VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Sledování pohybu osob ve vymezeném prostoru

Tracking the Movement of Persons Within the Defined Area

2018 Pavel Drábek

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Pavel Drábek**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Sledování pohybu osob ve vymezeném prostoru
Tracking the Movement of Persons Within the Defined Area

Jazyk vypracování: čeština

Zásady pro vypracování:

Cílem práce je implementovat algoritmus umožňující sledovat příchod a odchod osob z vymezeného prostoru. Předpokládá se využití charakteristických rysů jednotlivých osob pro ztotožnění příchozí a odchozí osoby např. pro účely stanovení počtu osob v objektu. Výsledkem práce bude jednoduchá konzolová aplikace umožňující zpracování zadané sekvence snímků nebo přímého výstupu z kamery. Aplikace bude realizována v jazyce C++ s využitím knihovny OpenCV. Je možné také využít dalších knihoven, např. pro detekci kostry člověka z RGB obrazu (OpenPose využívající GPU).

1. Seznamte se s algoritmy pro detekci a identifikaci osob.
2. Zhodnoťte možnosti identifikace osob z uvažovaných záběrů (tj. přehledový snímek chodby nebo místnosti z jedné či vícero kamer) a navrhněte řešení tohoto problému.
3. Navrhněte strukturu výsledného programu a zvažte využití vhodných knihoven pro řešení dílčích kroků detekce osob.
4. Vytvořte vhodné testovací sekvence včetně očekávaného výstupu (tzv. ground truth).
5. Otestujte funkčnost sytému na testovacích sekvencích.
6. Dosažené výsledky spolu s celým postupem práce pečlivě zdokumentujte v textové části.

Seznam doporučené odborné literatury:

[1] Wang , X., Zhao, R.: Person Re-identification: System Design and Evaluation Overview. pp. 351-370. 2014.
[2] Zhang, N. et al.: Frontal Faces: Improving Person Recognition Using Multiple Cues. CoRR. 2015.
[3] Cao, Z. et al.: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. CoRR. 2016.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Tomáš Fabián, Ph.D.**

Datum zadání: 01.09.2017

Datum odevzdání: 30.04.2018

doc. Ing. Jan Platoš, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 25. dubna 2018 .....

Rád bych na tomto místě poděkoval Ing. Tomáši Fabiánovi, PhD. za odborné vedení, cenné rady a připomínky při tvorbě této práce.

Abstrakt

Práce se zabývá sledováním pohybu osob ve vymezeném prostoru. Jedná se o úlohu opětovné identifikace, tedy dokázat identifikovat již dříve sledovanou osobu. Metoda předpokládá, že sledovaná osoba nezmění své oblečení, tudíž se hodí při opětovné identifikaci osoby v krátkém časovém úseku. Ke splnění úlohy je využívána knihovna OpenPose a konvoluční neuronové sítě. Trénování sítě probíhalo na datasetu MARS vytvořeného pro opětovnou identifikaci osob, který bylo nutné upravit s ohledem na použitou metodu. Práce popisuje navrženou metodu pro extrakci příznaků osoby v obraze. Z extrahovaných příznaků je sestavena matice příznaků o velikosti 36×20 pixelů jako vstup do konvoluční sítě. Značná část textu se věnuje návrhu a trénování konvoluční sítě, popisuje použité metody a podmínky pro výběr nejlepší architektury. Výstupem konvoluční sítě je deskriptor osoby, který systém použije pro nalezení nejbližší shody a odhadne totožnost osoby v obraze. Prezentované řešení dosáhlo na testovací sekvenci úspěšnosti 84,02%.

Klíčová slova: Opětovná identifikace osoby, Póza osoby, Konvoluční neuronová síť, ResNet, OpenPose

Abstract

This thesis deals with observations of people in a defined area. The main goal is their re-identification, in other words identifying a previously observed person. This method assumes, that the given person did not change their clothes during the observation, which means that it's mostly suitable in a short time period. The OpenPose library and convolutional neural network were used to successfully accomplish this task. The MARS dataset, which is tailored for a re-identification of people, was used with slight modifications for training of the neural network. This paper also describes a designed method for extracting features of people from given frame. These features were used to build a matrix of size 36×20 pixels and was used as an input to the convolutional neural network. Design and training of convolutional neural networks is greatly described in this paper, as well as what methods and conditions are necessary to choose the best possible type of architecture. The output of the convolutional neural network is a descriptor of given person, which the system uses to find the closest match and predict the identity of a person in the frame. The accuracy achieved in this solution was 84.02% using the testing set.

Key Words: Person re-identification, Human pose, Convolutional neural network, ResNet, OpenPose

Obsah

Seznam použitých zkratek a symbolů

Adam	–	Adaptive moment estimation
ANN	–	Artificial Neural Network
AVG	–	Average; průměr
CNN	–	Convolution Neural Network
CPU	–	Central Processing Unit; Procesor
DNN	–	Deep Neural Network
GPU	–	Graphics Processing Unit; Grafická karta
HDD	–	Hard Disk Drive; Pevný disk
ID	–	Symbol unikátně identifikující objekt nebo osobu
MARS	–	Motion Analysis and Re-identification Set
MAX	–	Maximum
ResNet	–	Residual Network
SGD	–	Stochastic gradient descent
SVM	–	Support Vector Machine

Seznam obrázků

1 Úvod

S průmyslovými kamerami se dnes může člověk setkat téměř na každém kroku. Hlídají firemní prostory i soukromé objekty, monitorují provoz na silnicích a sledují dění na veřejném prostranství. Pořízený záznam je mnohdy pouze ukládán a málokdy nastane situace vyžadující jeho zpracování. Takový případ může nastat, je-li potřeba nalézt podezřelou osobu napříč různými kamerami. Ruční zpracování tak velkého množství záznamu vyžaduje značné úsilí a je časově i psychicky náročné. Také je třeba počítat s faktorem lidské chyby, protože člověk sledující video může důležitý moment přehlédnout. Existuje technologie, která by dokázala zpracování zefektivnit a eliminovat chyby, kterých se člověk může dopustit?

Opětovná identifikace osoby je úloha rozpoznat osobu, která již byla dříve pozorována, napříč různými pohledy kamery a v různém čase. Jde o náročný úkol v oblasti strojového vidění, který může poskytnout užitečné nástroje pro aplikaci v bezpečnosti a video dohledu. Například sledování podezřelé osoby na více nepřekrývajících se kamerách nebo nalezení ztraceného dítěte na letišti. Uplatnění najde i v rozlehlých firemních prostorech, kdy je potřeba určit polohu zaměstnance.

Metody pro identifikaci člověka podle obličeje nebo otisku prstu jsou poměrně známé a prozkoumané. Možná je také identifikace podle duhovky oka či tvaru ucha. Při opětovné identifikaci se však tyto metody stávají často neúčinné, protože rozlišení obrazu je příliš malé a póza člověka příliš různorodá. Je tak často zapotřebí spolehnout se pouze na vzhled oblečení. Při popisu osoby se také využívá stavba těla a oděv se rozděluje podle jeho částí (např. trup a nohy). Z jednotlivých částí se následně získávají příznaky nízké úrovně (např. barva a textura). Vzhledem k řešené úloze klasifikujeme nejčastěji metody opětovné identifikace do dvou tříd — na základě videa a na základě obrázku. Zatímco u videa je možné pracovat například s trajektorií osoby nebo stylem chůze, u samotného obrázku přístup k dalším informacím nemáme. Tato práce se zabývá druhou třídou, tedy opětovné identifikace z obrazu. Metoda sice nemusí dosahovat takových výsledků, reálně se však dá využít ve více případech, protože není závislá na posloupnosti snímků.

Cílem této práce je seznámit se s algoritmy pro detekci a identifikaci osob. Na základě poznatků vytvořit vhodnou testovací sekvenci pro identifikaci osob v obraze včetně očekávaného výstupu a navrhnout strukturu výsledného programu. Implementovat algoritmus umožňující sledovat osoby ve vymezeném prostoru a na základě podobných rysů dokázat identifikovat opětovně příchozí osobu. Výsledná aplikace napsaná v C++ bude využívat knihovny OpenCV, dlib a knihovnu pro detekci pózy osoby OpenPose.

Práce je strukturovaná do tří částí. První část je teoretická, zabývá se problematikou, seznamuje čtenáře s pojmem konvoluční neuronová síť a popisuje její použití pro detekci a identifikaci osob. Na základě poznatků navrhuje vhodné řešení úlohy. Druhá, praktická část popisuje výběr vhodného datasetu a implementaci systému pro opětovnou identifikaci. Poslední část je zaměřena na nalezení nejlepší architektury konvoluční sítě pro implementovaný systém.

2 Přehled přístupů a metod

Počet prací zabývající se detekcí nebo opětovnou identifikací osoby se za posledních 10 let každoročně exponenciálně navyšuje [1]. Zkoumají člověka a zaměřují se na stavbu systému, který by dokázal porozumět jeho vzhledu, pohybu i aktivitám [2, 3]. Tato kapitola popisuje přehled přístupů a metod, které se zabývají problematikou opětovné identifikace nebo byly jiným způsobem přínosné pro tuto práci.

2.1 Detekce osoby

Metody pro detekci osob se dají rozdělit na dvě kategorie. Ta první vyžaduje separaci popředí od pozadí a následně jednotlivé objekty klasifikují do kategorií jako člověk, zvíře, vozidlo apod. Pozadí oddělují většinou z videa nebo využívají hloubkového obrazu. Druhý typ metod pracuje s obrazem přímo. Prohledávají obraz po částech, z jednotlivých oblastí získávají příznaky a následně rozhodují, zda se jedná o člověka či nikoliv.

Zástupcem první metody může být například Mixtura Gaussiánů [4], která reprezentuje každý pixel v obraze pravděpodobnostní funkcí určující jakých hodnot může nabývat pozadí. Pokud má pixel jinou hodnotu, jedná se o popředí. Do druhé třídy patří například Histogram orientovaných gradientů (HOG) [5]. HOG popisuje objekt pomocí distribuce gradientů v obraze, ty pak slouží jako příznaky pro klasifikaci pomocí SVM (Support Vector Machine).

Konvoluční sítě se ukázaly jako velmi účinné v oblasti analýzy obrazu a strojového vidění. Vyznačují se minimálním požadavkem na předchozí zpracování a dosahují vysokých přesností při detekci i klasifikaci objektů, včetně osob. Hlavními průkopníky jsou sítě R-CNN [6], Fast R-CNN [7], Faster R-CNN [8]. Jednotlivé sítě na sebe navzájem navazují a upravují konvoluční sítě pro detekci objektů. Pro R-CNN bylo potřeba nejdříve detekovat jednotlivé oblasti v obraze a následně je jeden po druhém posílat na vstup do předtrénované sítě AlexNet. Pro klasifikaci výstupu sítě byla použita SVM. Fast R-CNN bere na vstupu celý obraz a až následně klasifikuje regiony a softmax klasifikátorem namísto SVM. Faster R-CNN vyžaduje na vstupu pouze obraz. Z výsledných aktivačních map navrhne regiony a zároveň stejné aktivační mapy s navrženými regiony použije pro jejich klasifikaci.

2.2 Opětovná identifikace osoby

Zatímco detekce osoby je již poměrně prozkoumaná problematika, aktuální výzvu představuje identifikace osoby. Algoritmy pro opětovnou identifikaci se musí potýkat s mnoha obtížnými situacemi. Různě nasvětlená scéna může mít za následek změnu barevné škály. Osoby mohou být zachyceny z různých úhlu nebo se mohou otočit, což způsobí, že rozpoznávající elementy (kabelka, batoh, obrázek na tričku apod.), které mohly být po celou dobu vidět, jsou nyní zakryty. V neposlední řadě se může stát, že více různých osob je oblečných podobně nebo mají

stejnokroj. Ukázkovou výzvou pro opětovnou identifikaci osoby může být například situace na obrázku 1.

(a) stejná osoba (b) různé osoby

Obrázek 1: Ukázkový problém při opětovné identifikace osoby. (a) Snímky stejné osoby zabírají postavu z různých úhlů, jiného osvětlení a v různé kvalitě. Batoh na zádech není vždy vidět. (b) Různé osoby jsou oblečeny podobně a na první pohled nejsou od sebe rozeznat. Zdroj: [9]

V počátcích byly prezentovány převážně ručně zpracovávané algoritmy pracující s malým množstvím dat. V posledních letech však dochází k vývoji rozsáhlých datových souborů (datasetů) a hlubokých učících se algoritmů. Vzhledem k různým úlohám klasifikujeme nejčastější metody opětovné identifikace do dvou tříd - na základě obrázku a na základě videa. Zatímco u samotného obrázku nemáme k dispozici další informace, u videa je možné pracovat například s trajektorií osoby v průběhu času nebo stylem chůze.

V následujícím textu se často vyskytuje pojem top-X úspěšnost. Číslo X určuje do jaké pozice se musí objevit správná predikce, aby byla brána jako úspěšná. Takto definovanou úspěšnost je možné následně porovnávat napříč různými metodami a určit, která dosahuje lepších výsledků.

2.2.1 Histogram názvů významných barev

Práce [10] Y. Yang a dalších z roku 2014 představuje přístup pro identifikaci člověka založený na hledání významných barev v obraze. Definují 16 významných barev z RGB palety, které jsou zakódovány do jména barvy. Zakódováním barvy do názvu barvy tak mohou přiřadit více barev pod jeden název, čímž mohou zajistit invarianci barvy vůči osvětlení ve scéně.

Pro získání popisu osoby je výřez s danou osobou horizontálně rozdělen na šest částí a pro každou část je spočítán histogram názvů významných barev. Takto popsaná osoba se následně porovnává Mahalanobisovou vzdáleností za účelem správné identifikace. Tato metoda dosahuje úspěchu 37,8% při top-1 predikci, případně 68,5% u top-5 predikce na datasetu VIPeR [11]. Přestože se může zdát, že úspěšnost této techniky je mizivá, jedná se do té doby o jeden z nejlepších výsledků.

2.2.2 Siamská konvoluční síť

Jedna z prvních prací [12] z roku 2014, která pro identifikaci osoby použila hlubokou síť, představila „siamskou“ konvoluční síť. Na vstupu jsou dva obrazy a každý je vložen do jedné ze dvou stejně natrénovaných sítí. Každá z těchto sítí používá 3 vnitřní konvoluční sítě, které mají

sdílené parametry. Obraz je uvnitř horizontálně rozdělen na tři překrývající se části a každá část je vložena na vstup jedné ze tří vnitřních sítí. Každou z vnitřních sítí tvoří dvě konvoluční vrstvy s normalizační a MAX pooling vrstvou následovanou úplně propojenou vrstvou. Úplně propojená vrstva o 500 neuronech spojuje výstup všech tří sítí do jednoho vektoru velikosti 500. Tyto vektory z obou sítí jsou porovnány použitím kosinovy podobnosti (skalární součin), která rozhodne, zda se jedná nebo nejedná o stejnou osobu. Síť dosahuje top-1 úspěchu 34,49%.

2.2.3 Porovnání rozdílů v sousedních oblastí

Ze siamské konvoluční sítě vychází práce [13], která také používá dvě konvoluční sítě pro porovnání podobnosti dvou vstupních obrazů. Podobně má v každé síti dvě konvoluční vrstvy s maskami 5×5 a MAX pooling vrstvou. Liší se však ve spojující vrstvě obou sítí. Zatímco v práci [12] je použita úplně propojená vrstva spojující výstupy obou větví, zde byla navržena vrstva pojmenovaná Cross-input Neighborhood Differences.

Tato vrstva bere na vstupu obě výstupní aktivační mapy f_{25} a g_{25} s hloubkou 25 a produkuje dvě aktivační mapy K_{25} a K'_{25} s hloubkou 25, které na výstup spojí do jedné aktivační mapy s hloubkou 50. Každý pixel $K_i(x, y)$ se vypočítá podle vzorce

$$K_i(x, y) = f_i(x, y) * 5 * 5 - N[g_i(x, y)] \ , \tag{1}$$

kde i je index aktivační mapy v rozsahu $\langle 0, 25)$, (x, y) určují pozici pixelu v aktivační mapě a $N[g_i(x, y)]$ je výsledek konvoluce na aktivační mapě g_i pro pozici pixelu (x, y). Jelikož je tato operace asymetrická, aktivační mapa K'_{25} se spočítá obdobně, akorát se prohodí vstupní aktivační mapy f a g. Po této vrstvě následuje aktivační vrstva ReLU a další 2 konvoluční vrstvy zakončené úplně propojenou vrstvou rozměru 500, která na výstupních 2 neuronech predikuje, zda se jedná o stejnou nebo odlišnou osobu.

Takto vytvořená síť se stala s top-1 úspěšností 54,74% na datasetu VIPeR první hlubokou sítí v identifikaci osoby, která překonala tradiční metody.

2.2.4 Učení charakteristického vzhledu osoby

Další úspěšné použití konvoluční sítě pro identifikaci osoby je popsaná v práci [14]. Na rozdíl od předchozích prací požaduje na vstupu video nebo sekvenci snímků. Metoda spočívá ve výběru čtyř klíčových snímků osoby, které jsou vstupem do konvoluční sítě za účelem získání deskriptoru dané osoby. Pro každý snímek je jedna konvoluční síť skládající se z 5 konvolučních vrstev. Výstupy jednotlivých sítí jsou spojeny do jedné pomocí MAX poolingu, následují 2 úplně propojené vrstvy.

Klíčové snímky se vybírají tak, aby osoba na každém snímku byla v určité póze. K tomu použili detekci cyklu chůze, ze kterého následně určili 4 snímky, kdy má osoba nohy u sebe, nejvíce rozkročené a došlap a odraz levé nohy. Trénování probíhalo na datasetu iLIDS [15] s top-1 úspěšností 60,4%.

2.2.5 Segmentace a rozpoznání oblečení

Jeden z příznaků pro identifikaci člověka by mohl být popis oblečení, které má daná osoba na sobě. Segmentací a klasifikací oblečení se zabývají práce [16, 17, 18]. V práci [16] navrhli konvoluční síť podobnou VGG-16 (podsekce 4.3.3), kterou učí na vlastním datasetu FashionNet tak, aby predikovala orientační body, ze kterých následně predikuje i atributy. Orientační body slouží k ohraničení daného kusu oblečení, atributy pak k určení o jaký kus oblečení se jedná (tričko, mikina, sukně apod). Výsledná síť dosáhla top-3 úspěšností 82,58%, top-1 úspěšnost není uvedena. Metoda klasifikuje vždy jen jeden kus oblečení.

V práci [17] segmentují z obrazu jednotlivé kusy oblečení a následně navrhují podobné produkty z databáze čítající přes milion produktů. Nejprve odhadnou pózu osoby a grabcut algoritmem segmentují regiony, které jsou určeny k následnému zpracování. Každý region je klasifikován do kategorie (bunda, kalhoty, vlasy, kůže, ...) a z regionů obsahujících oblečení jsou získány příznaky pomocí jednoduchých metod namísto sofistikovaných z důvodu časové náročnosti. Vektor příznaků je sestaven kvantováním RGB barev a histogramem uniformních vzorů získaných lokálními binárními vzory (LBP) [19]. Pro nalezení podobných kousků oblečení je použita aproximace k-nejbližších sousedů. Nejedná se přímo o klasifikační problém, nicméně podobnost nalezených produktů je dostatečná k tomu, aby byl nalezený produkt z databáze použit jako příznak pro identifikaci osoby. Bohužel je tato metoda pomalá pro použití v systému pracující v reálném čase. Samotné hledání pózy zabere 1,7 sekundy, segmentace 0,7 sekundy a klasifikace 3,2 sekundy.

3 Návrh systému

Je žádáno, aby na vstupu programu bylo video nebo sekvence snímků, výstupem pro každý snímek pak predikce o jakou osobu se jedná. Predikce bude určena pouze z jednoho snímku, nebude zde tedy závislost na snímcích předchozích či následujících. Při spuštění programu bude pro opětovnou identifikaci potřeba dodat v požadovaném formátu informace o všech osobách, které se na záznamu vyskytují. Za validní informace bude považován jeden nebo více referenčních snímků celé osoby a její označení. Aby bylo možné osobu predikovat, musí být na snímku dostatečně viditelná a ve vhodném rozlišení. Následující text popisuje navržené řešení problému opětovné identifikace a popisuje systém a jeho fungování jako celek.

3.1 Použité technologie

Aplikace je realizována v jazyce C++ a využívá knihovny OpenCV [20] pro zpracování obrazu, Dlib [21] pro implementaci konvoluční sítě a OpenPose [22, 23, 24] pro detekci pózy osoby v RGB obrazu. Konvoluční síť je počítána na grafické kartě a knihovna OpenPose vyžaduje technologii CUDA [25]. Vývoj a testování systému probíhalo na následující konfiguraci:

- CPU: Intel Core i7-8700K, 3,7 GHz
- GPU: GeForce GTX 1080, 8 GB
- RAM: 16GB, 3 000 MHz
- OS: Windows 10 Pro, verze 1709, build 16299.371
- HDD: Seagate Barracuda, 2 TB, 7200 rpm

OpenCV je OpenSource knihovna navržená pro aplikaci v systémech z oblasti strojového vidění. Zaměřuje se na efektivní výkon. aby mohly systémy běžet v reálném čase. Obsahuje několik stovek algoritmů, které jsou kvalitně zdokumentované. Knihovna dlib je moderní C++ knihovna obsahující algoritmy pro učící se algoritmy. Dokumentace knihovny je ve formě okomentovaných zdrojových kódu při aplikaci konkrétního problému.

3.1.1 OpenPose

Knihovna OpenPose vznikla jako OpenSource projekt (pro výzkumné účely) spojením prací [22, 23, 24]. V navrženém systému slouží k detekci pózy jednotlivých osob v obraze. Vyznačuje se vlastností provést detekci v RGB obraze v čase nezávislém na počtu osob, dá se proto využít pro identifikaci osob v reálném čase. Detekce může běžet synchronně i asynchronně, knihovna má dostatečnou dokumentaci a neustále se vyvíjí. Navržený systém využívá OpenPose ve verzi 1.0, avšak během výzkumu vyšly další verze a nejnovější verze nese označení 1.3.

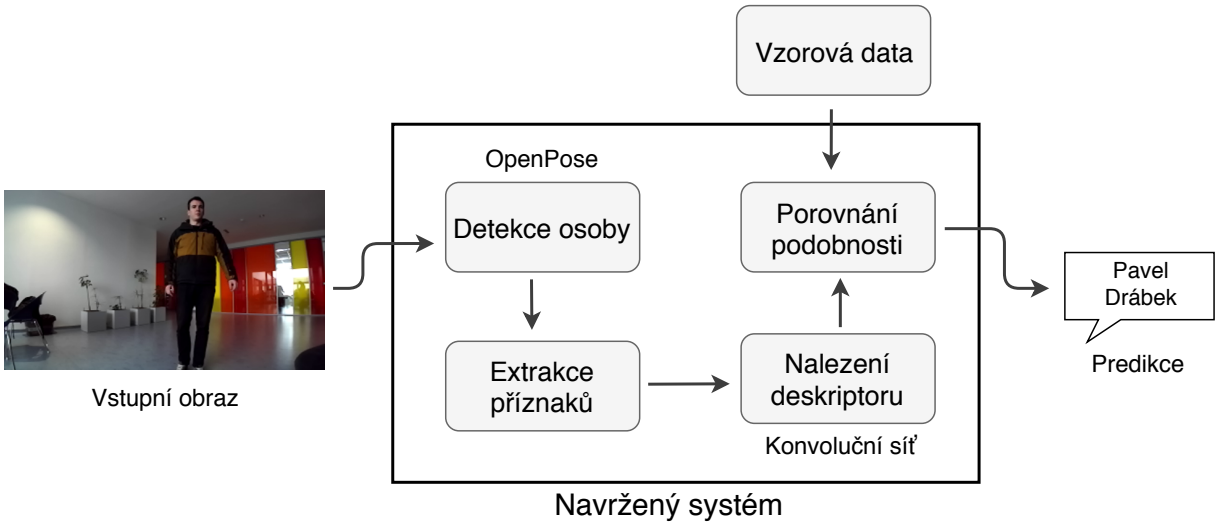
OpenPose detekuje 2D pózu libovolného počtu osob v obraze. Přístup využívá konvolučních sítí a částečných afinních polí pro detekci jednotlivých částí lidského těla, které následně přiřadí konkrétním osobám v obraze. Takto detekované a navzájem spojené body odpovídají hrubé stavbě lidského těla a jsou označovány jako póza osoby.

Obrázek 2: Nahoře: Odhad pózy knihovnou OpenPose, části těla stejné osoby jsou propojeny. Vlevo dole: Částečné afinní pole odpovídající končetině spojující pravý loket a pravé zápěstí. Barva odpovídá orientaci. Vpravo dole: Přiblížená predikce částečného afinního pole. Každý pixel v poli určuje pozici a orientaci končetiny. Zdroj: [22]

Jedná se o poměrně průlomovou práci v oblasti zpracování obrazu, na kterou během relativně krátké doby navázaly další práce [26, 27, 28]. Knihovna běží na grafické kartě využívající platformu CUDA [25] a framework Caffe [29]. V březnu 2018 byla přidána i podpora CPU.

3.2 Struktura systému

Systém se skládá z několika na sebe navazujících částí. Hrubý postup je velmi podobný postupům jiných prací popsaných v kapitole 2. V prvé řadě je nutné detekovat pozici osoby. Následně jsou získány příznaky, které budou reprezentovat danou osobu. Je požadováno, aby výsledný program dokázal identifikovat různé osoby, proto by predikce osoby neměla být založena na klasifikaci, nýbrž na získání popisu dané osoby (deskriptoru). Následným měřením podobnosti bude vybrán nejbližší možný kandidát.



Obrázek 3: Postup navrženého systému při identifikaci osoby.

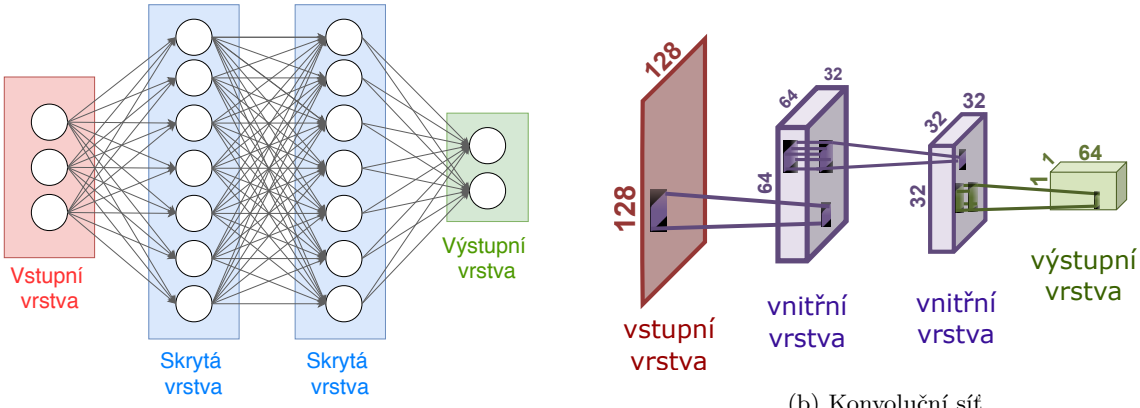
Na obrázku 3 je znázorněn model navrženého systému od zpracování vstupních dat až po výslednou predikci osoby. Pro detekci osoby je použita knihovna OpenPose, která zároveň určí klíčové body pro získání příznaků osoby. Pro získání deskriptoru osoby ze sestavených příznaků byla s ohledem na aktuální rozvoj a úspěchy použita konvoluční síť. Vstupem do této sítě je předzpracovaná matice příznaků, která je přímo závislá na detekované póze knihovny OpenPose. Výstupem konvoluční sítě je vektor příznaků popisující danou osobu. Tento vektor je porovnán se všemi referenčními daty a po nalezení největší shody je vrácena predikce osoby na snímku.

4 Konvoluční neuronové sítě

Konvoluční neuronové sítě (Convolution Neural Networks, ConvNets, CNN) se ukázaly jako velmi efektivní v oblasti strojového vidění. Úspěšně jsou používány pro identifikaci obličeje, dopravních značek a jiných objektů. V současné době je Konvoluční síť pravděpodobně nejlepší model v oblasti rozpoznání obrazu umělou inteligencí. Konvoluční neuronové sítě jsou modifikace neuronových sítí. Název získaly díky matematickému operátoru *konvoluce*, který ve svých výpočtech využívají. Následující text předpokládá, že je čtenář obeznámen s neuronovými sítěmi a jejich fungováním.

4.1 Architektura

Konvoluční sítě jsou stejně jako neuronové tvořeny neurony, váhami a postupně se jejich vrstvy zužují. Na vstupu je obraz a každý pixel odpovídá jednomu neuronu. Díky konvolučním a pooling vrstvám je možné z obrazu získat validní informace efektivněji. Na obrázku 4 můžeme vidět srovnání architektury jednotlivých sítí.



(a) Neuronová síť

(b) Konvoluční síť

Obrázek 4: Srovnání architektury Neuronové a Konvoluční sítě

Vrstvy neuronových sítí jsou jednorozměrné a se sousedními vrstvami jsou úplně propojené. Konvoluční sítě se liší v tom, že na vstupu očekávají trojrozměrnou matici a každý neuron ve vrstvě ovlivňuje následující vrstvu pouze ve svém okolí. Architekturu konvoluční neuronové sítě definuje:

- typy použitých vrstev
- počet vrstev
- pořadí vrstev
- parametry jednotlivých vrstev

4.2 Vrstvy Konvoluční sítě

Zatímco Neuronové sítě mají pouze jeden typ vrstvy, sítě konvoluční jich mají více a každá z nich má jinou úlohu. Přestože jednotlivých druhů vrstev není mnoho, je možné tyto vrstvy téměř libovolně kombinovat a měnit jejich hyperparametry. V závislosti na pořadí, počtu opakování a nastavených hyperparametrech dosahuje síť jiných výsledků (přesnosti), je různě veliká, komplexní a výpočetně náročná. Díky tomu existuje celá řada způsobů jakým je možné konvoluční neuronovou síť sestrojit. Hyperparametr je parametr, který se nastavuje před začátkem trénování a zůstává po celou dobu neměnný. Ostatní parametry se během učení mění.

4.2.1 Vstupní vrstva

Každá konvoluční neuronová síť začíná vstupní vrstvou. Její jediná úloha je převézt vstupní data do tensorů, se kterými bude síť dále pracovat. Je nutné, aby všechna vstupní data měla stejné rozměry, tedy šířku, výšku i hloubku. Hloubku vrstvy určuje počet kanálů v obraze. Pokud je tedy na vstupu standardní RGB obraz, má vstupní vrstva hloubku 3 (R, G, B). Na vstup je možno přivézt i obraz s větším počtem kanálů např. hloubkový obraz, který přidá ke třem kanálům ještě čtvrtý s hloubkovými daty. Naopak obraz ve stupních šedi bude mít hloubku jedna. Obecně je však možné do těchto kanálů přidat libovolná data jako třeba masku apod. Vstupní data by měly být vždy normalizovány v rozsahu $\langle 0, 1 \rangle$. Touto problematikou se bude více zabývat podsekce 4.2.5.

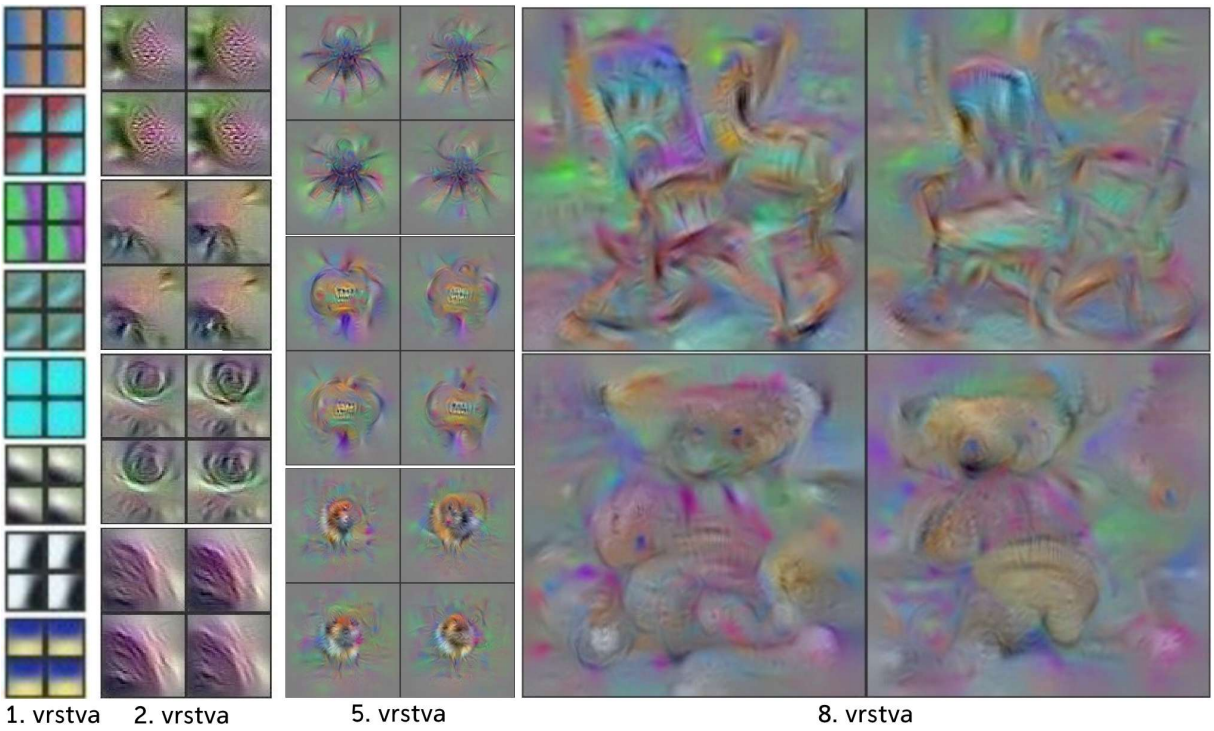
4.2.2 Konvoluční vrstva

Cílem konvoluční vrstvy je extrakce příznaků ze vstupního obrazu a zachovat jejich prostorový vztah. Jedná se o nejdůležitější vrstvu celé sítě, jelikož je to právě tato vrstva, která v sobě uchovává většinu natrénovaných parametrů. Provádí převážnou část výpočtů, proto se většinou jedná i o výpočetně nejnáročnější vrstvu. Hyperparametry konvoluční vrstvy jsou:

- počet konvolucí K, kde $K > 0$,
- rozměr konvolucí $W \times H \times D$, kde $W, H, D > 0$,
- velikost posunu S, kde $S > 0$.

Konvoluční vrstva se skládá z množiny učících se konvolučních filtrů. Každý filtr má tři dimenze, kdy šířka a výška jsou dány hyperparametry a většinou dosahují malých rozměrů, zato hloubka je vždy stejná jako hloubka obrazu předchozí vrstvy. Například první konvoluční vrstva může mít velikost $5 \times 5 \times 3$ (5 pixelů na šírku a výšku, 3 značí RGB kanály v obraze). Přestože rozměry konvoluční masky mohou být různé, na výstupu je vždy nový obraz s hloubkou jedna. Tomuto dvoudimenzionálnímu obrazu se říká aktivační mapa (také mapa příznaků), která reaguje na daný filtr v každé své pozici. Síť se tedy snaží najít takové filtry, které se aktivují jakmile uvidí v obraze určitý vizuální prvek jako třeba určitým způsobem orientovanou hranu, skvrnu určité barvy apod. Konvoluční vrstva provede K takovýchto konvolucí, každou s jiným

filtrem a výsledné aktivační mapy skládá za sebe. Tímto vzniká nový obraz s hloubkou K, který je zároveň výstupem konvoluční vrstvy.



Obrázek 5: Vizualizace konvolučních sítí; aktivační mapy. Zdroj: [30]

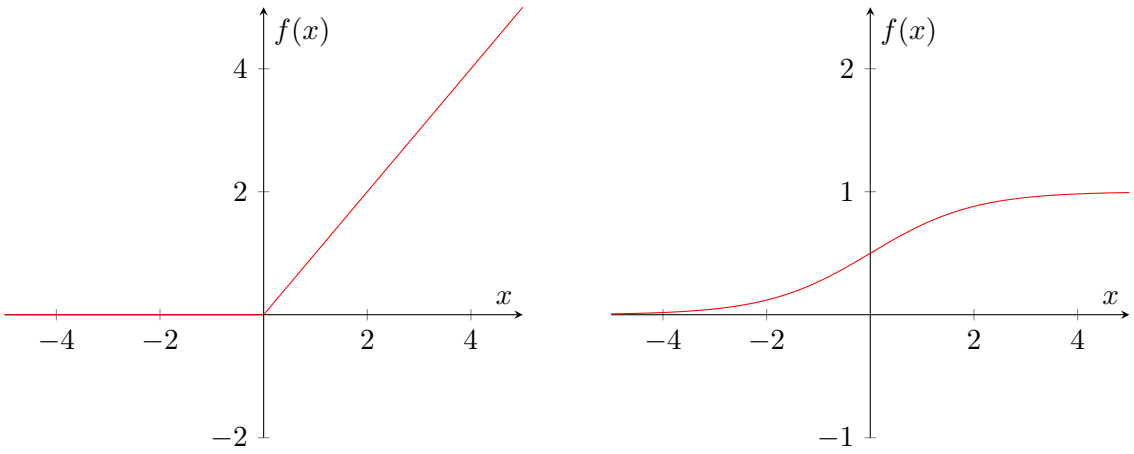
Vizualizaci aktivačních map konvoluční sítě můžeme vidět na obrázku 5. Čím hlouběji je konvoluční vrstva v síti, tím více hledá konkrétnější objekty. Filtry v první vrstvě hledají určitým způsobem orientované hrany. Na druhé vrstvě již skládáním orientovaných hran na sebe hledá hrany určitého tvaru, například kruhové objekty pro kolo nebo oko. Každá další vrstva hledá takové filtry, které z tvarů detekovaných předchozí vrstvou dokáží poskládat konkrétnější objekty. Na poslední 8. vrstvě už hledá konkrétní objekty jako například houpací křeslo nebo plyšového medvěda. Pro více informací o vizualizaci hlubokých sítí je doporučena práce [30].

4.2.3 Aktivační vrstva

Pokud by byly konvoluční sítě složené pouze z konvolučních vrstev, jednalo by se pouze o lineární systém [31]. Takový systém by dokázal řešit pouze omezený okruh problémů, který dnes hluboké sítě dokáží řešit. Aby tato linearita byla narušena, je zaváděna mezi jednotlivé konvoluční vrstvy aktivační vrstva.

Aktivační vrstva aplikuje na každý neuron vstupu aktivační funkci a má rozhodnout od jaké míry je hodnota neuronu užitečná a zda jej má „vypnout“. V popisu některých architektur se aktivační vrstva neuvádí z důvodu kratšího zápisu a někdy bývá aktivační vrstva zabudována přímo do konvoluční vrstvy. Existuje celá řada aktivačních funkcí, zde je uvedeno jen pár z nich:

- Sigmoid, funkce s předpisem $f(x) = \frac{1}{1+e^{-x}}$
- Rectified linear unit (ReLU); $f(x) = max(0, x)$
- Softmax; $f(x) = \frac{e^x}{\sum_{j=0}^{k} e^{x_j}}$, kde x je vektor



(a) funkce ReLU

(b) funkce Sigmoid

Obrázek 6: Srovnání průběhu funkce ReLU a Sigmoid.

Aktivační funkce ReLU transformuje záporné hodnoty na 0. Tato aktivační funkce je hojně využívaná převážně z důvodu její malé výpočetní složitosti. Softmax transformuje vstupní vektor x na vektor $f(x)$ při zachování dimenze. Složky výstupního vektoru jsou v rozsahu $\langle 0, 1\rangle$ a jejich součet je roven 1. Hodí se převážně pro klasifikační úlohy. Obor hodnot funkce sigmoid je v rozsahu $(0, 1)$ a využívá se pro predikci pravděpodobnosti.

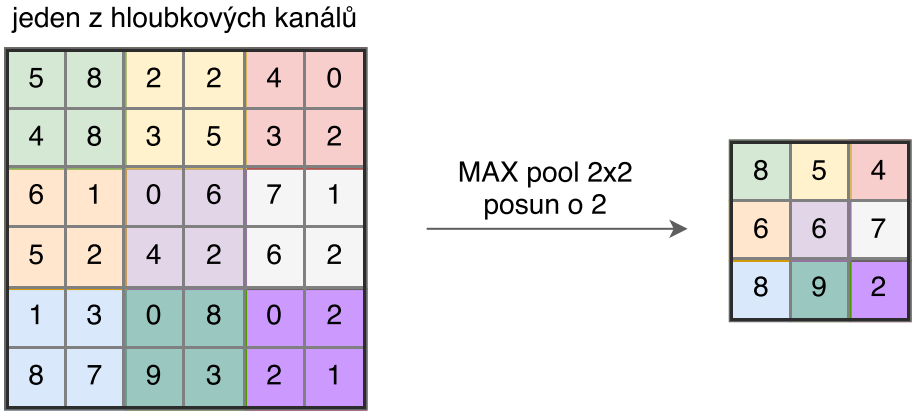
4.2.4 Pooling vrstva

Úlohou pooling vrstvy je zmenšit počet parametrů konvoluční sítě a tím urychlit její výpočet. Obvykle se opakovaně vkládá mezi jednotlivé konvoluční vrstvy. Vzorkování se aplikuje na každý hloubkový kanál vstupu a zmenšuje jeho rozměr v závislosti na níže uvedených hyperparametrech:

- rozměr filtru $W \times H \times D$, kde $W, H, D \geq 1$,
- velikost posunu S, kde $S > 0$.

Obvykle se volí filtr velikosti 2×2 s posunem o 2, což má za následek snížení počtu parametrů o 75%. Hloubka se většinou neudává, jelikož téměř ve všech případech se jedná o hloubku $h = 1$.

Obrázek 7 názorně ukazuje jakým způsobem probíhá vzorkování s operátorem MAX. Filtr vybere region o velikosti $W \times H$ a zapíše na výstup jedno číslo odpovídající nejvyšší hodnotě v regionu. Poté se posune o S na další region a postup opakuje, dokud neprojde celý vstup.



Obrázek 7: Příklad operace MAX pool velikosti 2×2 s posunem o 2.

Běžně používané operátory v pooling vrstvě jsou MAX, AVG a SUM. Zajisté budou existovat i další, většinou se však volí operátory s velmi jednoduchou matematickou operací z důvodu malého výpočetního výkonu.

Existují výzkumy snažící se navrhnout architekturu tak, aby nebyla závislá na vzorkovacích vrstvách. V práci [32] zjistili, že MAX pool vrstvy jdou jednoduše zaměnit za konvoluční vrstvu s vyšším posunem, aniž by klesla úspěšnost dané sítě. Síť složenou výlučně z konvolučních vrstev otestovali na datasetech CIFAR-10 [33], CIFAR-100 a ImageNet [34] a výsledky jsou konkurence schopné s jinými architekturami. Je tedy možné, že se v budoucnu budou navrhovat sítě bez pool vrstev.

4.2.5 Regulační vrstva

V rámci trénování se požaduje, aby konvoluční síť dosáhla nulové chyby - zero loss (více o loss vrstvě v podsekci 4.2.7). Během trénování však mohou nastat různé problémy. Například může síť dosáhnout nulové ztráty pouhým zapamatováním si celé trénovací množiny, což je nežádoucí jev, jelikož taková síť má potom slabé výsledky na testovací množině. Takto naučenou síť nazýváme přeučenou (overfitting). Druhým problémem může být, že síť nedokáže najít souvislost mezi vstupem a požadovaným výstupem. Takové trénování trvá příliš dlouho nebo se nemusí podařit vůbec. Těmto případům mají předcházet regulační vrstvy.

V případě přeučení sítě je potřeba trénovací množinu udělat hůře zapamatovatelnou. Toho lze dosáhnout buďto úpravou trénovací množiny nebo použitím regulační vrstvy Dropout. Dropout je velmi jednoduchá a účinná metoda, kdy je určité procento parametrů na vrstvě „zahozeno“. Síť tak dostane pokaždé jinou sadu dat a musí se naučit rozpoznávat podle více příznaků a znalosti tak rozdistribuovat po celé síti. Dropout se používá pouze při trénování a obvykle se požaduje, aby bylo zahozeno 30-50% dat.

Trénování sítě většinou trvá velmi dlouho (až v řádech týdnů). To je do jisté míry zapříčiněno změnou distribuce vstupu každé vrstvy během trénování. Pro řešení tohoto problému byla vyvinuta vrstva batch normalizace (normalizace dávky) [35]. Tato vrstva normalizuje vstupní

hodnoty, čímž přispívá k rychlejšímu trénování a vyšší efektivitě. Zároveň slouží jako regulační vrstva a v určitých případech nahrazuje vrstvu Dropout.

4.2.6 Úplně propojená vrstva

Úplně propojená vrstva (fully connected, FC) je výpočetní vrstva podobně jako vrstva konvoluční. Jedná se o stejnou vrstvu jakou mají sítě neuronové. Každý neuron je propojený s každým neuronem následující vrstvy. Nevýhoda této vrstvy je právě v počtu spojení, jelikož má za následek příliš mnoho parametrů – obzvlášť pokud následuje po konvoluční vrstvě. Doba učení se tak prodlužuje a síť má větší nároky na paměť. Většina moderních sítí se snaží tuto vrstvu nepoužívat a hledají alternativní řešení.

4.2.7 Loss vrstva

Během trénování je potřeba měřit jak dobře je konvoluční síť naučena. Loss funkce se obvykle implementuje jako samostatná vrstva a řadí se jako poslední vrstva sítě. Během trénování počítá chybu (loss) oproti ground truth. Algoritmus (optimalizátor), který zajišťuje trénování sítě, se snaží dosáhnout nulové chyby (zero loss) a tuto vrstvu využívá k ověření, zda jeho kroky vedou k požadovanému cíli.

Při navrhování sítě je nezbytně důležité uvědomit si, jaký typ problému chceme řešit a následně tomuto požadavku přizpůsobit loss funkci. Jednotlivé problémy můžeme dělit na klasifikaci a regresi:

- Klasifikace: Konvoluční síť přiřazuje objekty do předem definovaných tříd. Jedná se tedy o funkci, která zobrazuje spojitý vstup X na diskrétní výstup Y. Algoritmus, který provádí přiřazování (klasifikaci), se nazývá klasifikátor a jednotlivé třídy označujeme jako štítky (labels). Poslední plně spojená vrstva vrací vektor o stejné velikosti jako je počet klasifikačních tříd. Každá složka vektoru určuje pravděpodobnost s jakou náleží třídě odpovídajícího indexu.

- Regrese: Pokud je naším cílem rekonstrukce dat, lokalizace, nebo podobnost, pak je potřeba využít regresní funkci. Oproti klasifikaci je na výstupu spojitá funkce. Využívá se například pro lokalizaci daného objektu, takže na výstupu může být vektor velikosti 4 (x, y, šířka, výška) definující ohraničení. Při trénování odpovídá loss funkce L2 vzdálenosti mezi predikovaným vektorem a ground truth. Regresní konvoluční sítě se trénují na každý objekt zvlášť.

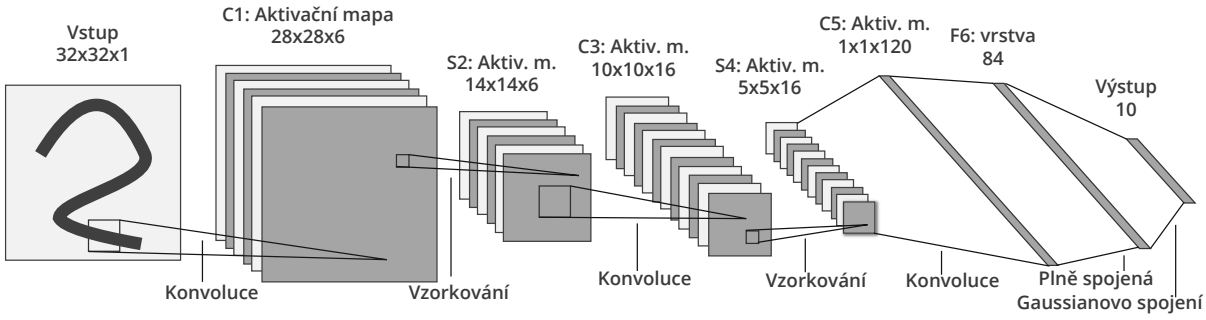
Není třeba zůstávat pouze u jedné loss vrstvy a je možné navrhnout i architekturu využívající loss vrstev více. To se často děje u spojených sítí, které navzájem spolupracují. Například jedna síť klasifikuje objekt, předá informace druhé síti a ta najde jeho pozici v obraze, viz Faster R-CNN [8].

4.3 Přelomové architektury

Za posledních 20 let se objevilo několik prací, které zásadně ovlivnily odvětví konvolučních neuronových sítí a hlubokého učení obecně. Jednotlivé sítě ukázaly nové způsoby návrhu architektury, možnosti optimalizace jednotlivých vrstev, případně které vrstvy jsou nahraditelné nebo redundantní. Tato část popisuje dle autorova uvážení ty nejzásadnější z nich a vysvětluje na jakých principech jsou založené a v čem jsou tak výjimečné. Většina z níže popsaných sítí porovnaly své výsledky v pravděpodobně nejprestižnější výzvě ILSVRC [34]. Výzva vznikla v roce 2010 a od té doby se každoročně opakuje.

4.3.1 LeNet

LeNet je první úspěšně aplikovaná konvoluční síť, kterou v roce 1998 popsali Yann LeCun a spol [36]. Tato síť se naučila rozpoznávat ručně psaný text a číslice s velmi malou chybou, zatímco nepotřebovala téměř žádné zpracování předem a předčila mnohé dosud zavedené techniky.



Obrázek 8: Architektura sítě LeNet-5 pro rozpoznávání čísel.

Architektura LeNet-5 na obrázku 8 vyžaduje na vstupu obraz ve stupních šedi s rozlišením 32×32 pixelů. V návrhu najdeme tři konvoluční vrstvy 5×5 a 2 pooling vrstvy spojené s aktivační funkcí sigmoid. Na výstupu klasifikační vektor o velikosti 10, kde každá složka vektoru odpovídá vzdálenosti k modelu čísla odpovídajícího indexu. Za pozornost stojí poslední konvoluce mezi vrstvami $S4$ a $C5$, kdy vstupní obraz má stejný rozměr jako konvoluční filtr. Výsledkem je obraz s rozlišením 1×1, který je možno jednoduše konvertovat na plně propojenou vrstvu.

Síť se v té době trénovala na počítači SGI Origin 2000 využívající jeden 200MHz procesor R10000 a trénování zabralo 2 až 3 dny.

4.3.2 AlexNet

Další sítí, která zásadním způsobem ovlivnila vývoj v oblasti rozpoznávání obrazu, je AlexNet [37]. Síť byla natrénována pro klasifikaci 1,2 miliónů obrázků do 1000 různých kategorií. Trénování probíhalo na datasetu ImageNet LSVRC-2010 a s obdobnou sítí vyhráli v soutěži

ILSVRC-2012 v top-5 testu první místo s chybovostí 15,3% (druhé místo dosáhlo chyby 26,2%). Tento úspěch obrovsky zvedl povědomí a zájem o konvoluční sítě.

Architektura sítě je vesměs jednoduchá, obsahuje 5 konvolučních vrstev (první z nich s filtrem o velikosti 11×11), MAX pooling, dropout a 3 plně spojené vrstvy. Její zvláštností je, že byla navržena pro dvě grafické karty GTX 580, protože paměť jedné karty 3GB nebyla dostatečná pro tak velkou síť. Tyto grafické karty si během učení vyměňovaly data pouze na vybraných vrstvách. Jako aktivační funkce byla implementována funkce ReLU, jelikož několikanásobně zkracuje čas trénování oproti funkci *tanh*. Aby nedocházelo k přeučení sítě, obsahuje vrstvu Dropout mezi jednotlivými plně spojenými vrstvami. Trénování na dvou grafických kartách trvalo 5 až 6 dnů.

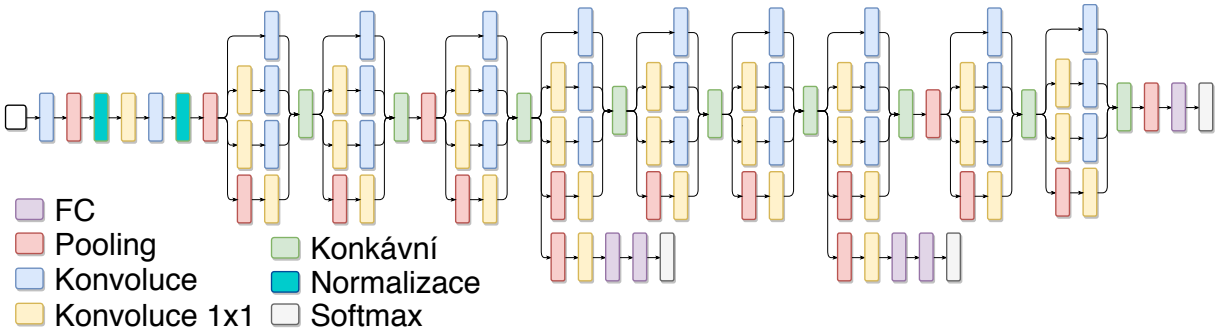
4.3.3 VGGNet

VGGNet [38] se zaměřila na zkoumání dopadu hloubky sítě na její úspěšnost. Bylo navrženo celkem 6 sítí, kdy nejúspěšnější z nich měla 16 vrstev. Aby bylo možné docílit takové hloubky, bylo potřeba snížit počet parametrů na vrstvu, čehož autoři dosáhli použitím konvolucí striktně o velikosti 3×3 a posunem o 1. V závislosti na hloubce konvoluční vrstvy v síti rostla i hloubka aktivační mapy. Zatímco první vrstva produkovala aktivační mapu $3 \times 3 \times 64$, aktivační mapa ve třinácté vrstvě už měla rozměr $3 \times 3 \times 512$. Celkově VGGNet dosahuje počtu 140 miliónů parametrů, přičemž více než dvě třetiny těchto parametrů generuje první plně spojená vrstva. Síť vznikla ještě v době, kdy se netušilo, že tyto vrstvy jdou nahradit, aniž by se snížila úspěšnost sítě.

Síť byla trénována na čtyřech grafických kartách Nvidia Titan Black přes 2 týdny. Přestože v soutěži ILSVRC-2014 dosáhla úctyhodného výsledku s chybou pouhých 7,3%, skončila na druhém místě.

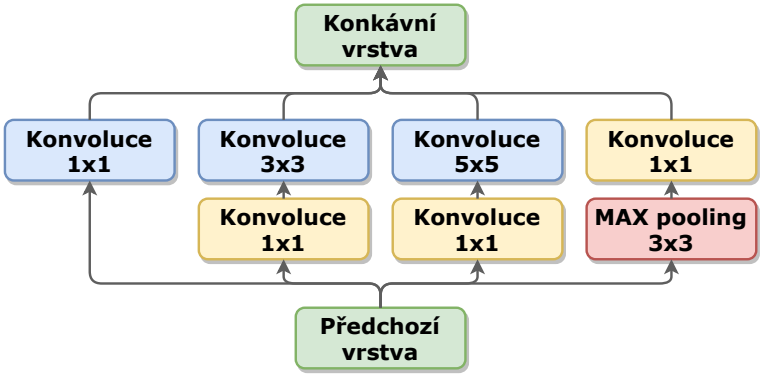
4.3.4 GoogLeNet

Byla to GoogLeNet [39], vyvinutá společnosti Google, která v soutěži ILSVRC-2014 předběhla VGGNet a skončila na prvním místě s chybou 6,7%. Zároveň se jedná o jednu z prvních konvolučních sítí, která přišla s jiným přístupem, než do té doby běžně používaným sekvenčním skládáním vrstev za sebe.



Obrázek 9: Architektura GoogLeNet.

Už z obrázku 9 je zřejmé, že architektura sítě je jiná než u doposud známých sítí. Síť je složena z 9 bloků, které jsou poskládány sekvenčně za sebe. Uvnitř jednotlivých bloků, které se nazývají Inception moduly [40], pak dochází k paralelním výpočtům. Inception moduly jsou základním stavebním prvkem této sítě, redukují počet parametrů sítě a přitom generují více dat pro učení. Vedlejší větve na obrázku 9 obsahují stejné vrstvy jako hlavní klasifikátor, slouží k rychlejšímu trénování a během predikce jsou odstraněny.



Obrázek 10: Inception modul v architektuře GoogLeNet.

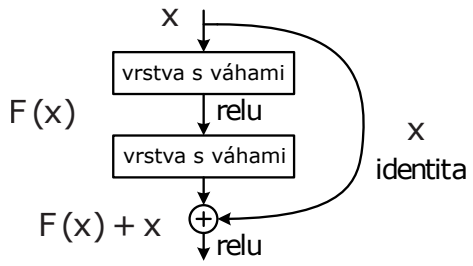
Inception modul na obrázku 10 odpovídá jednomu bloku z obrázku 9. Zatímco v jiných architekturách se rozhodovalo zda použít konvoluci nebo pooling, jakou zvolit velikost filtru a v jakém pořadí vrstvy volat, v GoogLeNet se rozhodli zavolat je všechny najednou a výstup jednotlivých větví spojit do jednoho hlubokého výstupu. Je potřeba si uvědomit, že čím více je vrstva v síti zanořena, je těžší k ní propagovat důležitá data, hůře se učí a naopak může snadněji docházet k přeučení sítě. Tím, že v Inception modulu jsou tyto vrstvy na stejné úrovni, mají přístup ke stejným informacím a snadněji mohou najít relevantní informace. Konvoluční síť tak neroste do hloubky, ale do šířky, což v praxi znamená, že má více dat, ze kterých se může učit.

Důležitou roli zde hrají i konvoluce 1×1. Ty jsou zde převážně z důvodu redukce hloubky vstupu. Pokud provedeme 20 konvoluci 1×1 na vstup $100 \times 100 \times 64$, zmenšíme hloubku aktivační mapy na $100 \times 100 \times 20$. Tím zůstane více místa v paměti pro další konvoluční vrstvy. Jelikož je tato transformace lineární a může tak docházet k přeučení, následuje za každou konvolucí 1×1 striktně nelineární aktivační vrstva ReLU.

Kromě Inception modulů snižuje počet parametrů velmi výrazně i AVG pool vrstva s filtrem 7×7, která redukuje výstup $7 \times 7 \times 1024$ konvoluční vrstvy na FC vrstvu $1 \times 1 \times 1024$. Ve výsledku má GoogLeNet pouze 4 miliony parametrů napříč více než 100 vrstvami, což se dá ve srovnání s AlexNet (60 milionů) a VGGNet (140 milionů) považovat za revoluci ve vývoji konvolučních sítí. Trénování probíhalo „na několika kartách během týdne“.

4.3.5 ResNet

Studie společnosti Microsoft při navrhování sítě ResNet [41] řešila problém, že čím je síť hlubší, tím je těžší tuto síť natrénovat. Dokonce ani nemusí nikdy dosáhnout takových výsledků jako méně hluboké sítě. Síť ResNet má však 150 vrstev (AlexNet pouze 8) a přesto se jí povedlo vyhrát ILSVRC-2015 s chybou 3,6%. Kromě hojného využívání batch normalizace popsané v podsekci 4.2.5 stojí za úspěchem tak hluboké sítě převážně reziduální bloky.



Obrázek 11: Reziduální blok sítě ResNet. Zdroj: [41]

Reziduální blok je postavený na sekvenci vrstva—ReLU—vrstva. Do vrstev s váhami patří například konvoluční vrstva nebo i pooling vrstva. Pokud takovou sekvenci označíme za funkci F, pak výstup z této vrstvy je $F(x)$ a výstup reziduálního bloku H zapíšeme předpisem $H(x) = F(x)+x$. V praxi to znamená, že za výstup takového bloku je přidán nezměněný vstup. To má za následek, že pokud se vrstvy reziduálního bloku nepodaří natrénovat na požadovaná data nebo by mělo dojít k přeučení, může být tato transformace v extrémním případě zahozena, jelikož jsou obsažena originální data a celý blok je tak „přeskočen“.

ResNet je doposud poslední konvoluční síť, která přišla s inovativním řešením. Aktuálně nejlepší výsledek v soutěži ILSVRC-2017 drží síť WMW [42] s chybou 2,25%, jedná se však „pouze“ o modifikovanou kombinaci sítí GoogLeNet a ResNet.

5 Implementace systému

Tato kapitola se zabývá implementací navrženého systému a jeho testováním. Jednotlivé sekce popisují výběr datasetu a důvody jeho filtrace, zvolený přístup pro sestavení matice příznaků a její rekonstrukci a vizualizaci deskriptorů osoby. Na modelovém příkladu bude prezentována vzniklá aplikace. Systém také využívá konvoluční neuronovou síť. Jejímu výběru a trénování se věnuje samostatná kapitola 6.

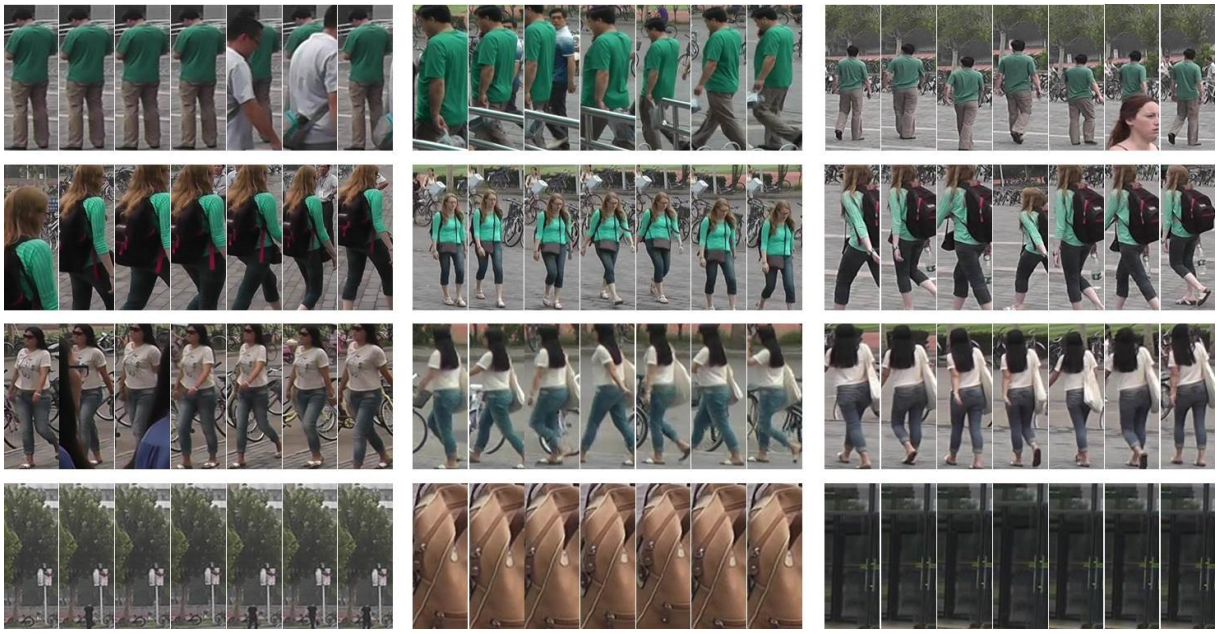
5.1 Použitý dataset

Pro určení funkčnosti a efektivity navrženého řešení bylo nutné sestavit množinu testovacích dat. Navržený systém může data k predikci získat pouze z RGB obrazu, je proto potřeba zajistit, aby pořízený obraz splňoval určité požadavky. Je nutné, aby na obraze byla celá postava, kterou je možné detekovat knihovnou OpenPose (více v podsekci 5.2.1). Scéna by měla být dostatečně nasvětlená a bez fyzických překážek, které by mohly sledovanou osobu jakýmkoliv způsobem zakrývat. Správným místem může být například vždy osvětlená chodba. Dataset by měl zároveň obsahovat pohled z více kamer.

Tvorba datasetu je časově i technicky náročná práce. Pro účely této práce byl vytvořen pouze malý dataset čítající v řádu stovek snímků od čtyř různých osob. Vlastní materiály vznikly za účelem zasvěcení do problematiky tvorby datasetu, následného stanovení kritérií pro výběr již existující datové sady a později k testování modelové situace. Za účelem opětovné identifikace však již vznikly rozsáhlé testovací množiny, proto se v tvorbě vlastního datasetu nepokračovalo.

5.1.1 Dataset MARS

Motion Analysis and Re-identification Set (MARS) [9] je dataset vytvořený organizací Springer. Pro záznam bylo použito 6 do jisté míry synchronizovaných kamer uvnitř kampusu univerzity Tsinghua v Číně. Jedna kamera natáčela v rozlišením 640×480 pixelů, zbylých pět pak v rozlišení 1920×1080. V datasetu je 1261 různých chodců, kdy každý je zaznamenán alespoň na dvou kamerách. Celkem je v datasetu 1 191 003 obrázků.



Obrázek 12: Ukázka z datasetu MARS. Zdroj: [9]

Díky vysokému rozlišení 128×256 jednotlivých osob je schopná knihovna OpenPose poměrně s velkou přesností detekovat pózu osoby. Dataset je strukturován na trénovací a testovací část. Každá část obsahuje množinu osob s různou identitou a obě části jsou vzájemně disjunktní. Pro každou identitu osoby pak existuje množina snímků ze dvou až šesti kamer. Jednotlivé osoby na snímcích jsou zachyceny většinou při chůzi a otočeny do různých směrů, což společně s různorodým oblečením zajišťuje dostatečně uniformní rozdělení trénovací množiny a natrénovaná síť tak má větší šanci obstát při nasazení do reálné praxe.

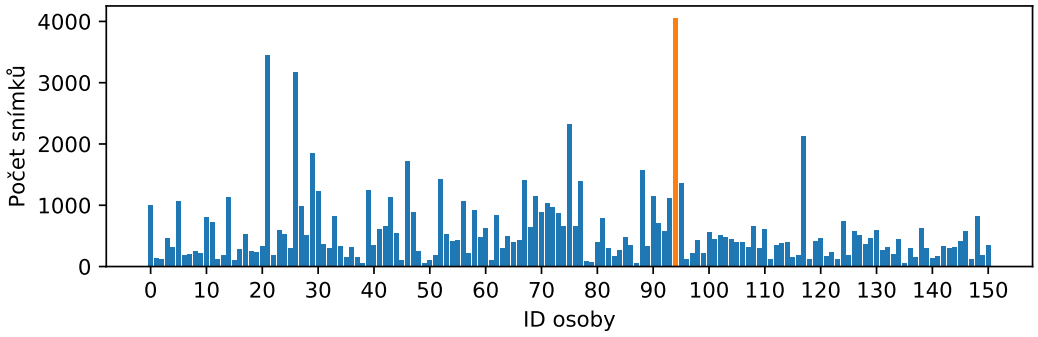
Obrázek 13: Vybrané snímky osoby s ID 0001 pro znázornění chyb v datasetu MARS.

Pro vytvoření datasetu byl použit algoritmus GMMCP [43] pro sledování trajektorie osoby. Kvůli velkému množství dat autoři neprovedli ruční validaci a proto dataset někdy obsahuje chyby. Většina chyb je způsobena křížící se trajektorií dvou nebo více osob. Jsou i případy, kdy algoritmus špatně určil ID osoby a snímek tak špatně zařadil. Jednotlivé chyby je možné vidět na obrázku 13. Z důvodu velkého množství dat nebyly tyto chyby v rámci práce opravovány.

5.1.2 Filtrace datasetu

Přestože většina snímků v datasetu splňuje požadavky stanovené výše, nachází se zde i značná část nevyhovujících snímků. Na mnoha z nich nebyla zachycena celá postava, jiné obsahovaly pouze detail na určitou část těla. Na některých snímcích nebyla žádná postava nebo naopak bylo postav více. Jelikož navržený systém pracuje s detekcí pózy osoby, jsou některé tyto snímky nevyhovující, protože by nebylo možné určit pózu a následně predikovat identitu osoby. Pro účely této práce bylo potřeba provést užší výběr datasetu MARS, konkrétně vznikly dvě datové množiny.

První množina vznikla za účelem trénování a testování konvoluční sítě. Snímky, u kterých nebyla detekována pouze část pózy (například pozice ruky nebo nohy), by mohly mít negativní vliv na trénování konvoluční sítě. Do této množiny proto byly zařazeny pouze snímky, na kterých je detekována jen jedna osoba, a u které je rozpoznána validní póza (viz podsekce 5.2.1). Druhá množina vznikla za účelem testování systému. Pro zařazení snímku do této množiny stačí, aby na něm byla detekována pouze jedna osoba. Snímek je schválen i pokud detekovaná póza není úplná, jelikož v reálném nasazení se může běžně stát, že určitá část nebude detekována.



Obrázek 14: Histogram vybrané datové množiny. ID 94 obsahuje 9 699 snímků.

Detekce pózy na snímku trvá v řádech desítek milisekund a z časových důvodů nebylo možné pracovat s celou množinou dat MARS. Pro první množinu tedy bylo vybráno prvních 151 osob u kterých bylo detekováno alespoň 150 snímků. V této množině je celkem 93 454 snímků. Histogram počtu snímků je na obrázku 14. Pro druhou množinu byly vybrány všechny snímky 30 osob, dohromady 19 648. Tyto dvě množiny obsahují různé osoby.

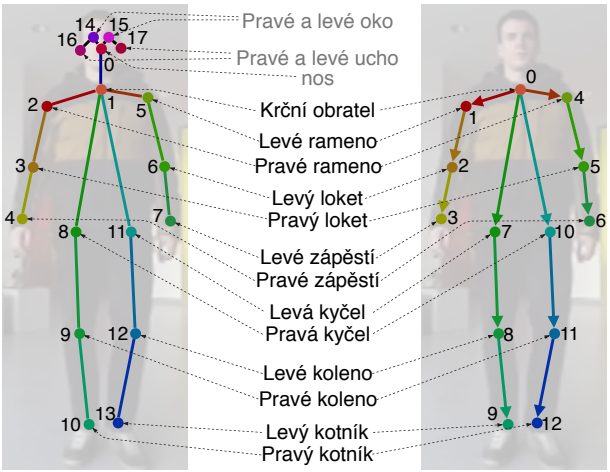
5.2 Extrakce příznaků

U konvolučních sítí popsaných v podsekci 4.3 stojí na vstupu předem nezpracovaný RGB obraz a následné příznaky pak hledá samotná konvoluční síť. Extrahování příznaků před vstupem do konvoluční sítě může tedy vypadat poněkud jako zbytečná práce. V návaznosti na práce detekující oblečení (viz sekce 2) je myšlenka tohoto postupu taková, že stačí pouze určitý vzorek oblečení, který může sloužit k identifikaci osoby. S ohledem na detekci pózy v obraze knihovnou

OpenPose se pak otevírají nové možnosti. Tato sekce popisuje jednu z těchto možností a její využití v navrženém systému. Extrakce příznaků probíhá stejně pro trénovaní sítě jako pro následnou predikci identity osoby.

5.2.1 Detekce pózy

Pro detekci pózy byla použita knihovna OpenPose popsáná v podsekci 3.1.1. Pro správnou detekci pózy je nutné zajistit obraz s dostatečným rozlišením. Za přijatelné rozlišení se dá považovat snímek, na kterém zabírá osoba 50×150 pixelů. Za určitého nastavení dokáže detekovat s velkou přesností i snímky s menším rozlišením, nicméně detekce takové pózy trvá již v řádů stovek milisekund, což u systému, který by měl běžet v reálném čase, není přijatelné (běžné nastavení detekuje pózu v obraze během 110 – 130 ms). Velice také záleží na osvětlení scény a pozici osoby.



Obrázek 15: Porovnání výstupní pózy OpenPose a pózy použité v systému.

OpenPose rozeznává 18 klíčových bodů postavy, viz obrázek 15. Pozici nosu, očí a uší však nelze určit, pokud je sledovaná osoba otočena zády ke kameře. Pro opětovnou identifikaci bylo použito pouze 13 bodů, které lze rozeznat z nejvíce možných úhlů, viz obrázek 15. Zároveň byla jednotlivým spojením (kostem) určena orientace pro pozdější využití při extrahování matice příznaků. V tuto chvíli je póza postavy převedena do vlastní datové struktury a není závislá na OpenPose. V případě potřeby je tak možné OpenPose nahradit jiným systémem pro odhad pózy bez nutnosti dalších úprav systému.

V reálném prostředí se může stát, že na sledovanou osobu nebude přímá viditelnost. Důvodem mohou být překážky ve scéně. Statické překážky v podobě dopravních značek, stromů, židlí apod. je možné do jisté míry eliminovat vhodným umístěním kamery. Automobil, zvířata, dveře, jiná osoba apod. patří mezi dynamické překážky vyskytující se v obraze náhodně a v závislosti na scéně je více problematické se takovýchto překážek zbavit. Knihovna OpenPose v určitých případech dokáže odhadnout pozici i částečně zakryté končetiny, přesto je pravděpodobné, že

detekce pózy sledované osoby za překážkou bude provedena špatně nebo nebude provedena vůbec.

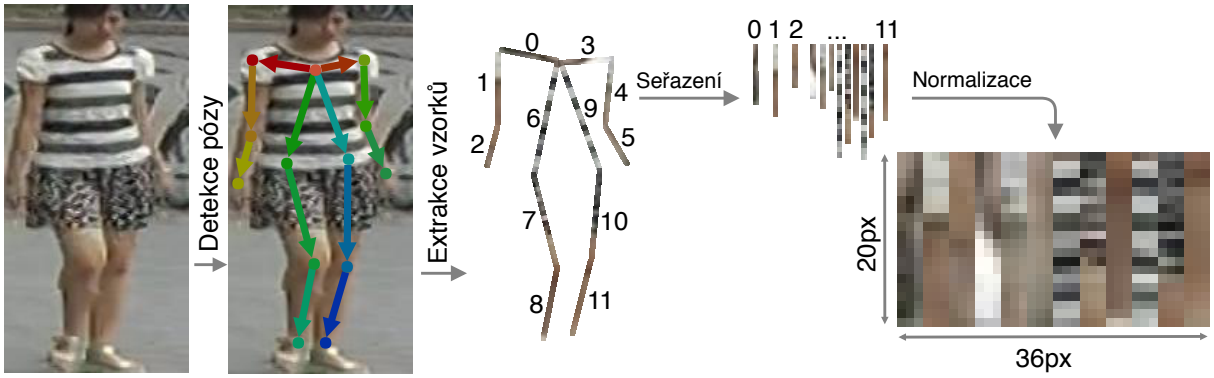
(a) (b) (c) (d)

Obrázek 16: Ukázka validní a neplatné detekce pózy. (a) Validní detekce. (b) Neplatná detekce zapříčiněná zakrytím osoby překážkou. (c) Neplatná detekce způsobena výřezem osoby. (d) Neplatná detekce způsobena překrytím dvou osob.

Jestliže není validně detekována póza člověka, je takový snímek označen jako neplatný a není možné pokračovat v identifikaci. Určitou výjimku představuje případ, kdy není detekována pouze jedna z párových končetin. Podrobněji je tato situace rozebrána v podsekci 5.2.4. Za validní detekci pózy se považuje rozpoznání všech 13 klíčových bodů pro sestavení 12 kostí, viz obrázek 15.

5.2.2 Matice příznaků

Jestliže je známa póza člověka, je možné jednoduše rozlišit důležité obrazové body od těch méně důležitých. Například pixely reprezentující okolí scény jsou pro identifikaci osoby nepodstatné. Dalo by se tvrdit, že jsou pro trénování sítě i zavádějící. Separací pozadí od postavy také můžeme zmenšit velikost vstupu do konvoluční sítě. Pokud zajdeme ještě dále, je možné ze sledované osoby extrahovat pouze vzorky oblečení a zmenšit tak vstupní obraz na minimum.



Obrázek 17: Proces pro sestavení matice příznaků

Jak avizuje obrázek 17, k extrakci vzorků oblečení je využita dříve detekovaná póza. Pro každou kost je z obrazu vytvořen výřez o rozměru $3 \times L$ px, kde L je délka detekované kosti. Každý z těchto výřezů je následně normalizován na rozměr 3×20. Naskládáním jednotlivých normalizovaných výřezů vedle sebe v předem daném pořadí vzniká matice příznaků dané osoby. Pořadí je definováno podle indexu kostí od nejmenšího po největší. Takto sestavená matice zároveň zachovává prostorový vztah příznaků. Výsledná matice příznaků má rozměr 36×20 a ve srovnání s původním obrazem o rozměru 128×256 je vstupní počet parametrů více než 45krát menší. Tak velký úbytek parametrů bude mít značně kladný dopad na dobu učení konvoluční sítě.

5.2.3 Vytvoření matice příznaků

Každá kost b je uspořádaná dvojice (p_1, p_2), kde p_1 je počáteční a p_2 koncový bod. Kost má svou velikost $L = |\overrightarrow{p_1 p_2}|$ a směr $\overline{v} = \overline{p_1 p_2}$. Výřez kosti g je obraz tvořený výběrem pixelů z původního obrazu f, který splňuje podmínku:

$$g(x, y) = f(p_{1x} + \overline{v} \cdot x, p_{1y} + \overline{v} \cdot y) \,, \qquad x \in \langle 0, 3), y \in \langle 0, L) \,. \tag{2}$$

Aby bylo možné vytvořit tento obdélníkový výřez kosti z obrazu, je nejprve nutné obraz otočit tak, aby byla kost ve svislé poloze. Je potřeba najít orientovaný úhel α, kdy otočením vektoru $\overline{v}$ o úhel α získáme vektor $\overline{u} = (0, 1)$. Vektor $\overrightarrow{w}$ je normálovým vektorem směrového vektoru $\overrightarrow{v}$ a platí pro něj předpis $\overrightarrow{w} = (v_y, -v_x)$. Úhel spočítáme podle předpisu:

$$\alpha = s \cdot \arccos(\overline{v} \bullet \overline{u}) \,, \qquad s = \begin{cases} 1, & \text{když } \overline{w} \bullet \overline{u} < 0 \\ -1, & \text{jinak} \end{cases} .$$

Pouhou aplikací rotace na obraz bychom rotovali kolem jeho středu a došlo by k transformaci kosti v obraze. Jednodušší varianta je tedy rotovat obraz kolem středu kosti, aby nedošlo k jejímu

posunu. Střed kosti $c(x, y)$ vypočítáme jako $c = p_1 + \frac{L}{2} \cdot \overline{v}$. Rotační matice je pak ve tvaru:

$$\begin{bmatrix} \cos\alpha & \sin\alpha & c_x(1-\cos\alpha) - c_y \sin\alpha \\ -\sin\alpha & \cos\alpha & c_x \sin\alpha + c_y(1-\cos\alpha) \end{bmatrix}.$$

Z otočeného obrazu f' je proveden výřez oblasti $C(x, y, \acute{s}i\check{r}ka, v\acute{y}\check{s}ka)$, kde C_x a C_y je pozice levého horního bodu výřezu a spočítá se vzorcem $C(x, y) = p_1 + (-1, 0)$. Výřez obrazu nyní odpovídá obrazu g a zbývá ho jenom normalizovat na rozměr 3×20. K tomu je využita OpenCV funkce `resize`. Celý postup je aplikován na všechny kosti a z jednotlivých normalizovaných výřezů je sestavena matice příznaků. V případě, že daná kost není rozpoznána, je chybějící část vyplněna nulovými hodnotami (černá barva).

5.2.4 Rekonstrukce matice příznaků

V případě, že některá z kostí není detekována, je ve výsledné matici nahrazena příslušná část černou barvou. Je tak učiněno z důvodu nutnosti zachovat stejný rozměr matice pro konvoluční síť. Takto poškozená matice však může zásadním způsobem ovlivnit tvorbu deskriptoru a následnou predikci osoby. V takovém případě jsou tři možnosti:

1. poškozená matice se i přes hrozbu špatné predikce pošle dál ke zpracování,
2. poškozená matice je zahozena a neposílá se k dalšímu zpracování,
3. poškozená matice se rekonstruuje a pošle se k dalšímu zpracování.

Rekonstrukce matice spoléhá na to, že jednotlivé kosti pózy jsou párové a oblečení osob vypadá stejně na levé i pravé straně. Pokud tedy nedetekujeme určitou kost v póze a k této kosti existuje validně detekována párová kost, je možné provést rekonstrukci této kosti. Rekonstrukce probíhá na matici příznaků a spočívá ve zkopírování oblasti patřící validně detekované párové kosti. V případě, že je kost špatně detekována a párová kost také, rekonstrukci nelze provést a predikce není provedena.

Obrázek 18: Porovnání matice příznaků před a po rekonstrukci.

Obrázek 18 ukazuje originální matici, záměrně poškozenou matici a zrekonstruovanou matici. Tyto matice byly převedeny na deskriptor a vzdálenost deskriptorů mezi originální a poškozenou

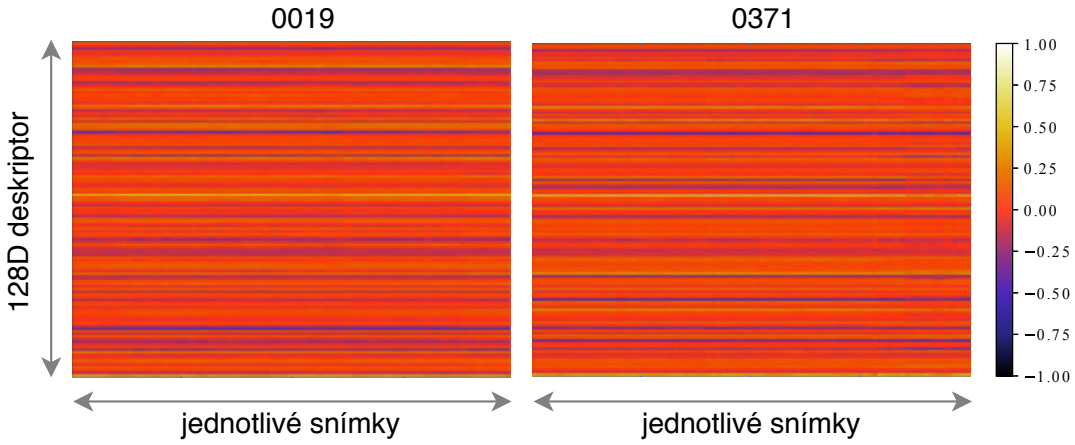
maticí je 0,540. Takto vzdálené deskriptory mohou být v určitých případech predikovány jako odlišná osoba. Vzdálenost mezi deskriptory originální a zrekonstruované matice je vzdálenost 0,164. Jedná se o poměrně dobrý výsledek, viz sekce 5.3. Hlavní důvod rekonstrukce je však navýšit počet možných predikcí v obraze. Tabulka 1 porovnává počet možných predikcí před a po rekonstrukci. V testovací sekvenci je detekováno celkem 16 185 osob a díky rekonstrukci je možné provést o 36% více predikcí.

Tabulka 1: Navýšení počtu možných predikcí po rekonstrukci matice příznaků.

Detekcí celkem	Před rekonstrukcí	Po rekonstrukci
16 185	8 120	13 947
100%	50,17%	86,17%

5.3 Deskriptor osoby

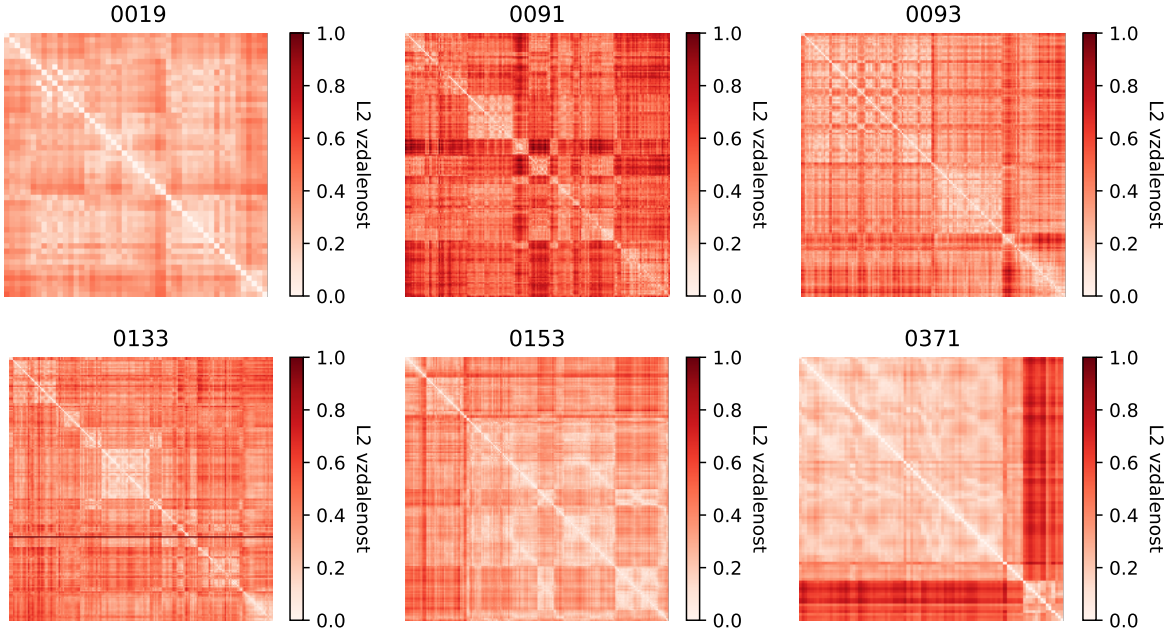
Deskriptor osoby určitým způsobem popisuje osobu na snímku a systém na základě dvou deskriptorů určuje, zda se jedná o stejnou osobu či nikoliv. Deskriptor je tvořen vektorem 128 hodnot a pro jeho získání byla použita konvoluční neuronová síť. Návrhem, trénováním i testováním se zabývá celá kapitola 6. Na obrázku 19 jsou vizualizovány deskriptory pro osoby s ID 0019 a 0371 datasetu MARS. Deskriptory vytvořila síť Net-05 popsaná v sekci 6.4.2. Svislá osa reprezentuje jeden deskriptor a na vodorovné ose jsou zaznamenány jednotlivé deskriptory pro všechny snímky osoby z upraveného datasetu.



Obrázek 19: Vizualizace deskriptoru pro osoby 0019 a 0371. Je žádoucí, aby deskriptory stejné osoby byly podobné. Deskriptory různých osob by měly být odlišné.

Je žádoucí, aby stejné osoby na snímcích byly reprezentovány stejnými nebo vzájemně blízkými deskriptory a naopak rozdílné osoby měly deskriptory vzájemně vzdálené. Podobnost deskriptoru je spočítána jako euklidovská vzdálenost rozdílu obou vektorů. Obrázek 20 vizualizuje vzájemnou podobnost deskriptorů všech snímků vybraných osob. Pro každou osobu je

sestrojena jedna matice. Řádky i sloupce odpovídají indexům snímku a hodnota v buňce odpovídá vzdálenosti mezi těmito dvěma snímky.



Obrázek 20: Vizualizace podobnosti deskriptorů stejné osoby. Náhodně vybráno 6 osob.

Z matice pro osobu s ID 0371 můžeme vyčíst, že deskriptory lze rozdělit na dvě množiny. V každé množině jsou si deskriptory navzájem podobné avšak napříč množinami jsou od sebe vzdálené. V případě osoby s ID 0091 už je situace horší. Téměř všechny deskriptory jsou si vzájemně vzdálené. Z vybraných snímků těchto dvou osob na obrázku 21 můžeme pozorovat, že oblečení osoby s ID 0371 vypadá jinak zepředu než zezadu. Tento problém lze vyřešit určením dvou referenčních snímků pro danou osobu. U osoby s ID 0091 můžeme vidět, že je oblečená celá v bílém a na první pohled není vidět nic, kvůli čemu by měly být jednotlivé deskriptory rozdílné. Pozdější pozorování ukázalo, že konvoluční síť má problém s rozlišením lidí s oblečením bílé barvy. Tento problém nebyl vyřešen.

(a) 0091 (b) 0371

Obrázek 21: Snímky stejných osob se vzdálenými deskriptory.

6 Návrh a testování konvoluční sítě

Konvoluční síť je součástí navrženého systému pro opětovnou identifikaci člověka. Tato kapitola vysvětluje, jakým způsobem byla navrhována architektura finálně použité sítě. První sekce popisuje práci s daty a důvody použití určitých komponent. Následně je popsán způsob trénování a testování neuronové sítě. Poslední část je experimentální, zabývá se konkrétním návrhem různých sítí a důvodem výběru jednotlivých vrstev a jejich pořadím.

6.1 Práce s daty

Konvoluční sítě jsou často využívány pro klasifikaci tisíce různých objektů v nezpracovaném obraze [37, 38, 39]. Systém navržený a popsaný v této práci však pracuje na vstupu s vyextrahovanou maticí příznaků popsanou v podsekci 5.2.2 a na výstupu je požadován deskriptor predikované osoby. Vstupem je tedy RGB obraz s rozlišením 36×20 a na výstupu vektor čítající 128 hodnot.

6.1.1 Předzpracování datasetu

Detekce pózy knihovnou OpenPose trvá v řádu desítek milisekund. Při trénování je tedy prakticky nemožné vytvářet matici příznaků pokaždé znovu. Pro účely trénování proto bylo nutné zpracovat dataset do tvaru, který obsahuje pouze matice příznaků. Nový dataset má stejnou hierarchii i stejné pojmenování souborů, aby bylo možné přiřadit matici příznaků k původnímu nezpracovanému obrázku. Při trénování jsou tak načítány přímo matice příznaků a není potřeba provádět opětovnou detekci pózy.

6.1.2 Ground truth

Ground truth pro trénování i testování zajišťuje hierarchie vstupních dat a jejich načítání. Jednotlivé osoby jsou rozmístěny do složek podle jejich identity. Vstupem pro načtení dat je cesta ke složce, ve které jsou tyto data obsažena. Načítaná data jsou ukládána do datové struktury `std::vector<std::vector<T>`, kdy vnitřní vektor obsahuje všechny snímky jedné osoby, index vnějšího vektoru pak určuje o jaké ID osoby se jedná. V rámci ground truth se porovnávají pouze indexy vnějšího vektoru.

6.2 Trénování

Konvoluční sítě obsahují parametry (váhy), které se podílejí na výsledné predikci sítě. Počet parametrů je závislý na architektuře a různé nastavení hodnot těchto parametrů ovlivňuje výslednou úspěšnost konvoluční sítě. Správné nastavení vah není triviální. Jedná se o NP úplný problém a vyzkoušet všechny kombinace je časově nemožné. Proto se konvoluční sítě stejně jako neuronové sítě trénují. Tato sekce popisuje postup při trénování konvoluční sítě, jaké metody byly pro trénování použity a co vedlo k použití těchto metod. Pro testy byly použity sítě s

jednodušší architekturou z důvodu rychlejšího testování a občas byly zopakovány na složitější architektuře pro ověření správnosti. Současně byl pro učení z důvodu křížové validace dataset rozdělen na 120 osob pro trénování a 30 osob pro testování.

6.2.1 Log

V průběhu trénování jsou průběžně zaznamenávány různé hodnoty, které se po skončení trénování zapíšou na disk do několika souborů pro následnou analýzu. Všechny názvy souborů mají prefix T tvořený časovým razítkem (čas spuštění trénování), aby mohl být soubor jednoznačně identifikován:

`T_architecture.txt` Obsahuje popis architektury konvoluční sítě. Pořadí a hyperparametry jednotlivých vrstev.

`T_network.dat` Serializovaná natrénovaná síť ve stavu po končení trénování.

`T_network_best.dat` Serializovaná natrénovaná síť vybraná v průběhu trénování jako nejlepší kandidát podle podmínky popsané v kapitole 6.2.6.

`T_training.csv` Záznam z průběhu trénování je ukládán do CSV souboru. V každém řádku jsou uloženy následující hodnoty:

step počet provedených kroků od začátku trénování,

loss chyba na trénovacích datech v daném kroku,

train přesnost naměřená na trénovacích datech,

test přesnost naměřená na náhodném výběru z testovacích dat, který se mění při každém měření,

testStatic přesnost naměřená na náhodném výběru z testovacích dat, který zůstává v průběhu trénování neměnný,

timestamp časové razítko v době zápisu.

`T_trainObjs.dat` Seznam osob použitých pro trénování sítě

`T_testObjs.dat` Seznam osob použitých pro testování sítě v průběhu trénování

`T_log.txt` Obsahuje dodatečné informace o trénování, např. hodnoty hyperparametrů, podmínky pro zastavení sítě apod.

6.2.2 Mini-batch

Dat pro natrénování sítě je příliš mnoho, než aby se je všechny podařilo načíst do paměti počítače. Konvoluční síť tedy bylo potřeba trénovat po malých dávkách, které při každém kroku vyměníme. Metoda se označuje jako mini-batch. V paměti jsou tak drženy pouze cesty k souborům namísto

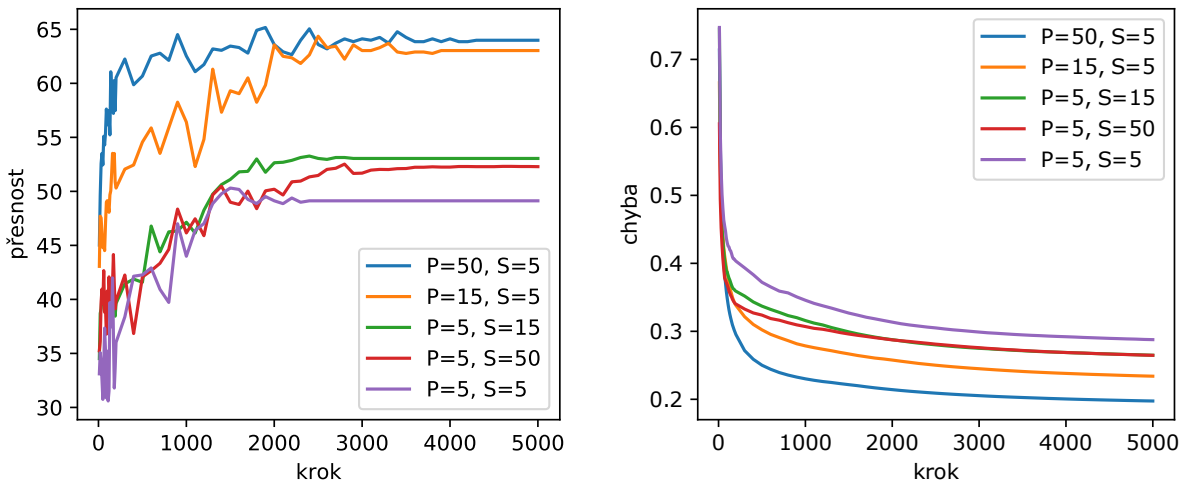
celých snímků a v případě potřeby jsou načteny. Jelikož takové trénování neprobíhá po epochách na celých datech, ale každý krok obměňuje úzký výběr dat, je potřeba aby se optimalizační algoritmy hledající minimální chybu orientovaly podle průměrné chyby namísto aktuální.

Pro zajištění rovnoměrného výběru vzorků v průběhu trénování, byl pro každý mini-batch vybrán náhodný P počet unikátních osob a od každé osoby náhodný S počet vzorků. Celkový počet vzorků C pro trénování daného kroku vypočítáme jako $C = P \cdot S$. Výběrem pokaždé stejně velkého počtu vzorků od dané osoby se tento způsob stává odolný vůči nepoměrnému zastoupení celkového počtu snímků pro každou osobu v datasetu.

Tabulka 2: Architektura CNN pro testování parametrů mini-batch.

#	**Název vrstvy**	**Parametry**	**Posun**
1	Konvoluční + ReLU	$256 \times 3 \times 3$	$1, 1$
2	MAX Pool	2×2	$2, 2$
3	Konvoluční + ReLU	$256 \times 3 \times 3$	$1, 1$
4	Globální AVG Pool	Vše do jednoho	-
5	Plně propojená	128	-

Úspěšnost a doba trénování závisí na zvolených hodnotách P, S a výsledné hodnotě C. Pro nalezení závislosti mezi těmito dvěma parametry a průběhem trénování bylo nutné provést sérii testů. Architektura konvoluční sítě použila pro testování parametrů mini-batch vychází ze sítě LeNet a je zaznačena v tabulce 2. Výsledky testů byly zpracovány do grafů na obrázku 22 a tabulky 3. Pro každý test bylo v rámci trénování provedeno 5 000 kroků a byla provedena dvě měření. Na obrázku 22 je zaznačena průměrná hodnota z obou testů.



Obrázek 22: Přesnost (vlevo) a chyba (vpravo) sítě v průběhu trénování pro rozdílné parametry P a S pro výběr vzorků mini-batch. Hodnoty v grafu ukazují průměrnou hodnotu z obou měření.

Z výsledků testů vyplývá, že celkový počet vzorků C má vliv na úspěšnost trénování a největší vliv na úspěšnost má parametr P, tedy počet unikátních osob v každé dávce. Naopak počet vzorků stejné osoby S nemá na úspěch příliš velký vliv a zároveň prodlužuje dobu trénování více než parametr P.

Tabulka 3: Výsledky testování parametrů P a S pro výběr vzorků mini-batch. Pro každý test byly provedena dvě měření #1 a #2.

Parametr P	**50**	**15**	**5**	**5**	**5**
Parametr S	**5**	**5**	**5**	**15**	**50**
#1 Přesnost natrénované sítě [%]	61,32	61,21	51,71	49,21	49,08
#2 Přesnost natrénované sítě [%]	67,23	60,00	44,96	52,67	56,80
#1 Nejmenší ztráta	0,200	0,236	0,295	0,274	0,270
#2 Nejmenší ztráta	0,195	0,232	0,280	0,256	0,259
#1 Čas trénování [s]	164	68	37	89	248
#2 Čas trénování [s]	162	67	36	86	246

6.2.3 Optimalizace načítání dat pro mini-batch

Trénování probíhá na grafické kartě, zatímco načítání dat zajišťuje procesor. Aby grafická karta nemusela čekat než procesor načte data a výkon grafické karty tak byl využit naplno, byly jednotlivé snímky pro mini-batch načítány paralelně. Konkrétně bylo vyhrazeno pět vláken pro neustálé načítání dat do fronty. Nakonec zůstal nejslabším článkem pevný disk, který z důvodu malé čtecí rychlosti nestíhal číst data. I přes tento nedostatek se však podařilo dosáhnout většího využití výkonu grafické karty a tento krok se ukázal jako správný. Tento způsob načítání byl použit už při testování mini-batch v předchozí kapitole.

6.2.4 Loss vrstva

Výstupem konvoluční sítě není klasifikace, ale deskriptor osoby. Pro výpočet chyby je použita funkce, která zohledňuje podobnost mezi deskriptory jednotlivých osob. Je žádoucí, aby deskriptory stejného člověka si byly navzájem podobné a naopak deskriptory různých lidí navzájem odlišné. Knihovna dlib již implementuje třídu `loss_metric`, která chybu na množině deskriptorů umí spočítat. Chyba E se počítá vždy pro celou dávku dat jako součet chyby e_{ij} pro každé dva deskriptory i a j. Tento vztah se dá zapsat vzorcem:

$$E = s \sum_{i=0}^{N} \sum_{j=i+1}^{N} e_{ij} \ , \tag{3}$$

kde N je celkový počet snímků v dávce a s je měřítko, které dává chybě vypovídající hodnotu. Je závislé na počtu kombinací snímků obsahující stejnou osobu. Protože máme stejný počet snímků

S pro každou osobu, je možné použít vzorec:

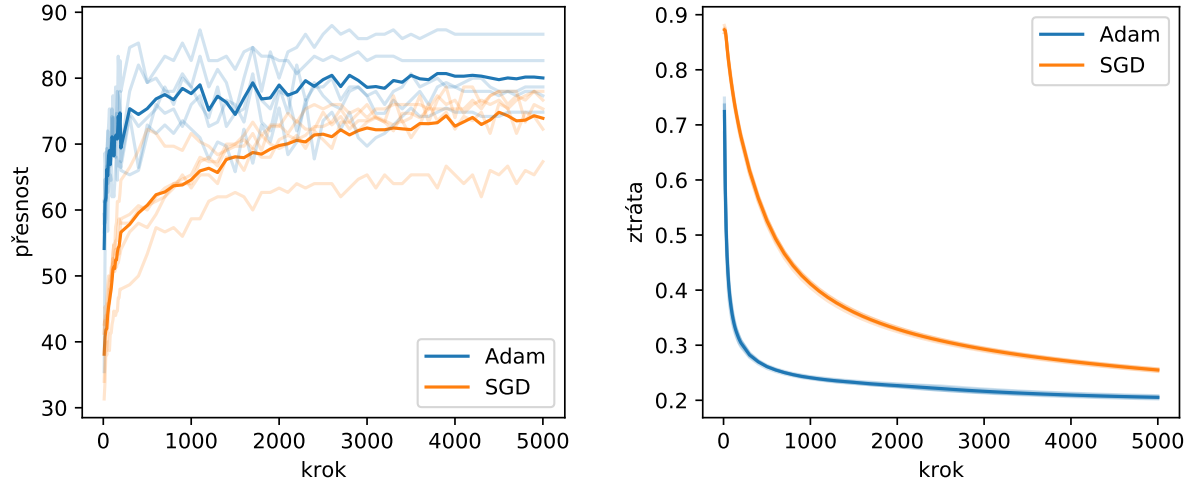
$$s = \frac{1}{N(S-1)} \,. \tag{4}$$

Chyba e_{ij} mezi dvěma deskriptory pro osoby l_i a l_j je závislá na podmínce, zda jsou tyto osoby stejné ($l_i = l_j$) a vzdálenosti deskriptorů d_{ij}. Chyba se pak odvíjí od hranice (threshold) t, která určuje jak moc mají být deskriptory různých osob vzdálené. Zároveň je potřeba stanovit volnost hranice (margin) m, aby vzdálenosti všech deskriptorů nebyly limitně u hranice t za účelem nulové chyby. Tento vztah určuje vzorec:

$$e_{ij} = \begin{cases} max(0, d_{ij} - t + m) & l_i = l_j \\ max(0, t - d_{ij} + m) & l_i \neq l_j \end{cases} \,. \tag{5}$$

6.2.5 Výběr optimalizátoru

Pro výběr optimalizátoru byly provedeny podobné testy jako pro výběr parametrů mini-batch a byla použita stejná architektura sítě reprezentovaná tabulkou 2. Pro každý optimalizátor bylo provedeno 5 měření a výsledky testů jsou prezentovány v tabulce 4 a grafech na obrázku 23, kde je pro větší přehlednost zanesena i průměrná naměřená hodnota. Z výsledků testů vyplývá, že optimalizátor Adam rychleji konverguje k menší chybě i vyšší přesnosti. Čas potřebný k trénování 5 000 kroků nebyl volbou optimalizátoru ovlivněn. Vzhledem k výsledkům testu byl ve většině experimentů použit optimalizátor Adam.



Obrázek 23: Přesnost a chyba sítě v průběhu trénování pro optimalizátory Adam a SGD. Pro každý optimalizátor bylo provedeno 5 měření (světlá barva). Sytou barvou je zaznačena průměrná hodnota všech měření.

Tabulka 4: Výsledky testování optimalizátoru SGD a Adam. Pro každý optimalizátor bylo provedeno 5 měření.

	Přesnost natrénované sítě [%]	**Nejmenší ztráta**	**Čas trénování [s]**
Adam	74,71	0,208	154
	80,00	0,202	155
	73,87	0,205	154
	83,20	0,207	154
	83,20	0,203	153
SGD	70,39	0,253	156
	71,60	0,257	156
	73,20	0,252	157
	77,80	0,257	157
	80,27	0,255	156

6.2.6 Výběr nejlépe naučené sítě během trénování

Natrénovaná síť po ukončení trénování nemusí dosahovat těch nejlepších výsledků. Proto je během trénování síť průběžně testována a pamatuje si stav sítě s nejlepší přesností na testovacích datech, tzv checkpoint. Test v plném rozsahu trvá příliš dlouho (viz sekce 6.3), proto je pro testování v průběhu trénování také použita metoda mini-batch (podsekce 6.2.2).

Během trénování jsou po určitém počtu kroku prováděny vždy dva na sobě nezávislé testy. Jeden test probíhá vždy na nově vybrané množině dat (dynamický test), pro druhý test se data v průběhu trénování nemění (statický test). Pokud je přesnost vyšší než doposud nejvyšší přesnost pro daný test, je proveden checkpoint. Po ukončení trénování je pro každý test vrácena síť ve stavu, kdy byl proveden poslední checkpoint. Je požadováno, aby takto vrácená síť měla větší nebo stejnou přesnost než síť ve stavu po ukončení trénování (žádný test).

Předpokládá se, že s rostoucím počtem dat v testovací množině bude test více vypovídat o přesnosti sítě. Následující série testů má za cíl prozkoumat jaký vliv mají mini-batch parametry P a S na kvalitu výběru checkpointu během trénování. Hledáme takové parametry, které sestaví testovací data tak, aby příliš neprodlužovala dobu trénování a výsledky testu během trénování co nejvíce odpovídaly finálnímu testu (viz sekce 6.3). Parametry P a S pro testovací mini-batch nemají vliv na průběh trénování, ovlivňují pouze výběr checkpointu.

Pro každé hodnoty parametrů bylo provedeno pět měření. U každého měření a hodnoty parametrů je označena síť s nejvyšší přesností. Všechna měření byla prováděna na stejné architektuře reprezentované tabulkou 2. Výsledky testů jsou zpracovány v tabulkách 5 a 6.

Tabulka 5: Výsledky testů parametrů mini-batch pro testování během trénování. Pro každý test bylo provedeno 5 měření. Světle zeleně je označena nejvyšší úspěšnost v rámci měření. Tmavě zeleně a červeně pak nejlepší a nejhorší průměrná úspěšnost v rámci testu.

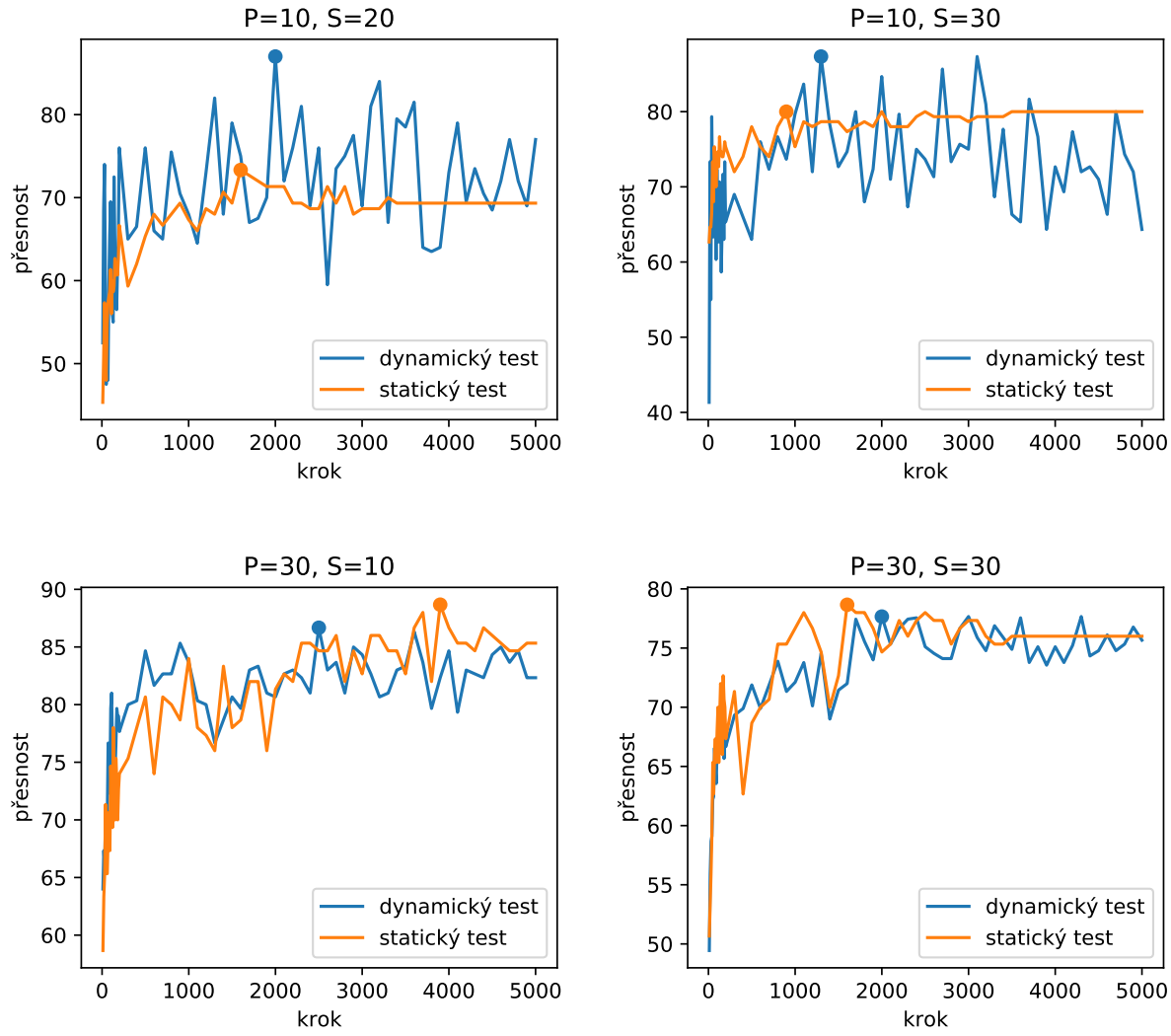
			#1	#2	#3	#4	#5	Průměr
Test. množina	**P**	**S**	**Přesnost [%]**					
žádná			81,12	85,00	83,40	80,08	83,16	82,55
proměnlivá	10	10	82,80	82,40	81,16	87,68	84,24	83,74
statická			83,96	81,56	83,08	87,60	83,08	83,86
žádná			78,71	82,47	77,00	79,80	77,93	79,18
proměnlivá	20	10	80,19	79,20	81,73	77,87	79,40	79,68
statická			81,16	79,73	83,60	78,93	78,53	80,39
žádná			83,16	83,56	78,64	82,48	83,40	82,25
proměnlivá	30	10	83,00	81,72	83,32	85,20	86,96	84,04
statická			82,52	82,80	82,44	79,72	87,32	82,96
žádná			86,33	80,00	81,47	85,53	78,13	82,29
proměnlivá	10	20	84,33	83,73	81,60	84,27	78,80	82,55
statická			81,40	81,87	80,60	82,27	80,40	81,31
žádná			83,60	81,40	83,53	81,40	83,06	82,60
proměnlivá	10	30	83,80	87,07	84,73	79,67	80,40	83,13
statická			83,53	78,47	77,00	79,13	80,53	79,73
žádná			83,20	82,52	81,16	85,16	83,32	83,07
proměnlivá	30	30	81,32	83,24	82,52	83,32	85,92	83,26
statická			81,96	80,48	78,68	82,32	84,96	81,68

Tabulka 6: Doba trénování [s] 5 000 kroků v závislosti na parametrech testovací mini-batch.

Parametr P	**10**	**20**	**30**	**10**	**10**	**30**
Parametr S	**10**	**10**	**10**	**20**	**30**	**30**
#1	154	156	157	158	158	168
#2	156	155	156	156	157	168
#3	155	157	157	156	156	168
#4	154	156	157	155	157	169
#5	154	156	157	155	159	168
Průměr	154,6	156	156,8	156	157,4	168,2

Z výsledků vyplývá, že pro výběr checkpointu na proměnlivé testovací množině je potřeba, aby C (celkový počet dat testovací množiny) byl větší nebo roven 200. Zároveň od počtu 300 je síť vybraná statickým testem horší než síť po ukončení trénování. Test neprokázal, který z parametrů P a S má větší vliv na úspěšně vybraný checkpoint. Výsledek, že sítě vybrané

statickým testem dosahují nejhorší přesnosti, byl překvapivý, proto byl průběh trénování pro parametry $(P, S) = \{(10, 20), (10, 30), (30, 30), (30, 10)\}$ ve třetím měření zanesen do grafů na obrázku 24. V grafu je kromě výsledků průběžných testů také zaznačeno, kdy došlo k výběru checkpointu. Z grafů $(10, 20), (10, 30), (30, 30)$ na obrázku 24 vyplývá, že checkpoint pro statický test se vytvořil v době, kdy má síť ještě tendenci se učit a proto vybraná síť dosahuje nejhoršího výsledku. V grafu pro hodnoty parametrů $(30, 10)$ naopak vidíme případ, kdy přesnost statického testu ještě roste, zatímco dynamický test již stagnuje. Dynamický test má tedy vyšší vypovídající hodnotu o průběhu trénování sítě. Je také vidět, že testovací množiny s vyšší hodnotou C mají více „stabilní“ průběh. To znamená, že výsledky testování přesnosti jsou více odolné vůči náhodnému výběru a jsou tak více vypovídající o aktuálním naučení sítě. Parametr P má na vypovídající hodnotu testování větší vliv než parametr S.



Obrázek 24: Průběh trénování sítě se zaznačeným checkpointem pro různé hodnoty parametrů P a S. Zaznamenané hodnoty jsou ze třetího měření.

6.3 Testovaní

Testování na malém vzorku dat je využíváno při trénování sítě pro určení přesnosti na trénovací a testovací množině a po ukončení trénování určuje úspěšnost sítě na celé množině testovacích dat. Za tímto účelem byl stanoven jednotný způsob jakým bude testování probíhat.

6.3.1 Log

Obdobně jako u trénování jsou i při testování zaznamenávána data, která se následně ukládají na disk v podobě několika souborů. Názvy souborů obsahují prefix N odpovídající názvu testované sítě a prefix T tvořený časovým razítkem (čas spuštění testování).

`N_T_confmat.csv` CSV soubor obsahující hodnoty pro vytvoření konfúzní matice.

`N_T_confmat.png` Vizualizovaná konfúzní matice knihovnou OpenCV.

`N_T_labelNames.txt` Seznam osob použitých pro testování sítě v průběhu trénování.

`N_T_log.txt` Obsahuje dodatečné informace o testování (počet testovaných osob, počet vzorků od každé osoby, úspěšnost testování)

6.3.2 Referenční deskriptor

Jelikož výstupem konvoluční sítě není klasifikace ale deskriptor, není možné přímo určit pravdivost predikce. Aby bylo možné deskriptoru přiřadit unikátní identifikátor osoby (ID), je nejprve nutné přiřadit pro každé ID jeden deskriptor, který bude brán jako referenční. Pro účely trénování a testování sítě byl tento referenční deskriptor určen náhodně před zahájením testování z množiny testovacích dat. V případě, že máme určenou referenční množinu, měříme pro každý výstupní deskriptor L2 vzdálenost oproti každému deskriptoru v referenční množině. Výstupnímu deskriptoru je poté přiřazeno stejné ID jako má jemu nejbližší deskriptor z referenční množiny. V tuto chvíli je deskriptor klasifikován a je možné určit, zda proběhla predikce správně či nikoliv na základě ground truth.

6.3.3 Konfúzní matice

Jednotlivé predikce jsou zanášeny do konfúzní matice. Jedná se o specifické uspořádání dat do matice, umožňující vizualizaci úspěšnosti klasifikačních algoritmů. Každý řádek matice reprezentuje instance predikované třídy a každý sloupec instance skutečné hodnoty. V matici je pak možné pozorovat, které třídy síť nedokáže dostatečně rozlišit. Je žádoucí, aby hodnoty na diagonále matice byly co největší a hodnoty mimo diagonálu nulové. V tuto chvíli již má deskriptor přiřazen identifikátor, a tak je možné predikci v závislosti na ground truth zaznamenat do konfúzní matice pro další zpracování. Úspěšnost sítě je počítána podle vzorce pro senzitivitu:

$$senzitivita = \frac{TP}{TP + FN} \, . \tag{6}$$

TP určuje počet správných predikcí (hodnoty na diagonále) a FN určuje počet nesprávných predikcí (hodnoty na mimo diagonálu). Po každém testu sítě je konfúzní matice ukládána na disk pro možnosti dalšího zpracování a zpětného ověření výsledků. Pro vizualizaci konfúzní matice byl vytvořen skript v jazyce Python. Ukázku takové matice je možné vidět na obrázku 28 v sekci 6.4.2.

6.3.4 K-násobná křížová validace

Při testování hlubokých sítí je potřeba zhodnotit jak dobře se síť dokáže naučit z dostupných dat. Testování na datech použitých při učení má velmi malou vypovídající hodnotu, jelikož síť si může zapamatovat trénovací množinu a dojde k přeučení. Pro testování je tedy potřeba použít jiná data než při trénování. Zároveň je potřeba přihlédnout k situaci, že máme poměrně malou množinu dat. V takových situacích je možné použít metodu křížové validace. Princip spočívá v rozdělení testovací množiny na k částí a vyjmutí jedné části, která se použije pro trénování. Postup se k-krát opakuje a výsledek se zprůměruje. Pro testování sítí byla testovací množina rozdělena na 5 částí.

iterace 1	iterace 2	iterace 3	iterace 4	iterace 5
testování	trénování	trénování	trénování	trénování
trénování	testování	trénování	trénování	trénování
trénování	trénování	testování	trénování	trénování
trénování	trénování	trénování	testování	trénování
trénování	trénování	trénování	trénování	testování

Obrázek 25: Rozdělení datového souboru při k-násobné křížové validaci (zde k=5).

6.4 Návrh architektury konvolučních sítí

Po seznámení s konvolučními sítěmi bylo potřeba navrhnout vlastní architekturu. Tento proces zabral v rámci práce pravděpodobně nejvíce času, jelikož mnoho sítí po hodinách strávených návrhem, trénováním a testováním skončilo neúspěchem a bylo potřeba začít znovu. Jednotlivé sítě byly inspirovány sítěmi AlexNet a ResNet. Celkem bylo provedeno okolo 800 testů na přibližně 100 sítích. Proces je shrnut do následujících dvou experimentů. Jednotlivé sítě v rámci testování byly pojmenovány podle časového údaje ve kterém vznikly, pro text této práce byly vybrané sítě přejmenovány v rámci přehlednosti.

6.4.1 Experiment 1

Cílem prvního experimentu bylo ověřit si znalosti nabyté teorií v praxi a navrhnout funkční architekturu sítě, která bude použita v systému popsaném v kapitole 5. V rámci experimentu bylo vytvořeno přes 40 architektur, které se lišily hloubkou i pořadím jednotlivých vrstev. V

následujícím textu jsou popsány architektury, které v rámci experimentu dosáhly nejlepšího výsledku.

Tabulka 7: Výběr navržených architektur v rámci prvního experimentu.

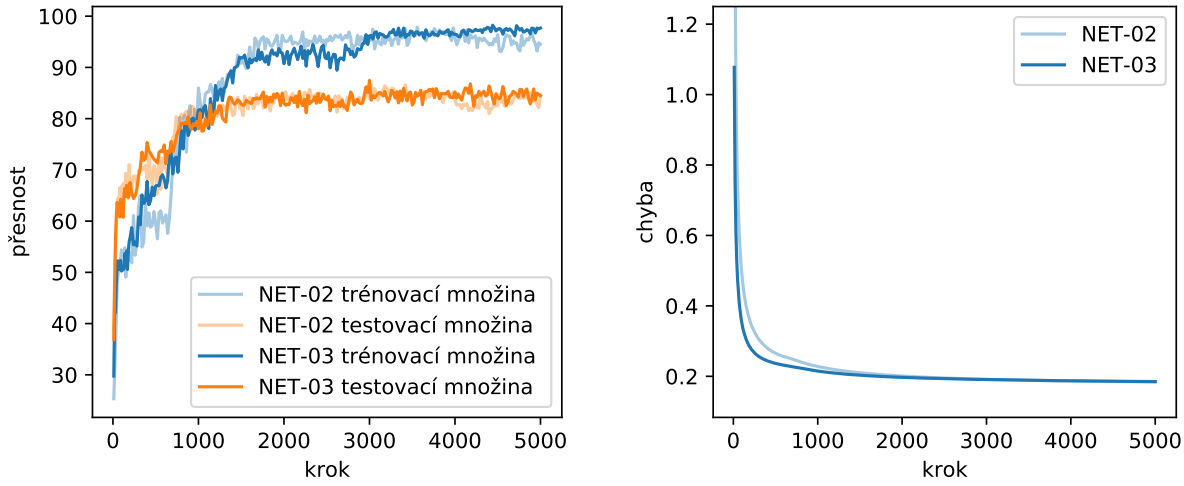
#	**Net-01**	**Net-02**	**Net-03**
1	Konvoluce 128 @ 3 × 3 / 1	Konvoluce 128 @ 3 × 3 / 1	Konvoluce 128 @ 3 × 3 / 1
2	ReLU + BN	ReLU + BN	BN + ReLU
3	Dropout	AVG Pool 2 × 2 / 2	AVG Pool 2 × 2 / 2
4	AVG Pool 2 × 2 / 2	Konvoluce 128 @ 3 × 3 / 1	Konvoluce 128 @ 3 × 3 / 1
5	Konvoluce 128 @ 3 × 3 / 1	ReLU + BN	BN + ReLU
6	ReLU + BN	AVG Pool 2 × 2 / 2	AVG Pool 2 × 2 / 2
7	AVG Pool 2 × 2 / 2	Konvoluce 128 @ 3 × 3 / 1	Konvoluce 128 @ 3 × 3 / 1
8	Konvoluce 128 @ 3 × 3 / 1	ReLU + BN	BN + ReLU
9	ReLU	Globální AVG Pool	GlobálníAVG Pool
10	Globální AVG Pool	Plně spojená 128	Plně spojená 128
11	Plně spojená 128	-	

S ohledem na poměrně malé rozlišení vstupního obrazu byl volen menší počet konvolučních vrstev s naopak větším počtem konvolucí. Poslední Globální pooling vrstva byla použita pro propojení aktivační mapy s plně propojenou vrstvou. Sítě byly trénovány na datasetu čítající 150 osob, viz kapitola 5.1, parametry mini-batch byly nastaveny na hodnoty $(P, S) = (50, 5)$. Úspěšnost jednotlivých architektur můžeme vidět v tabulce 8.

Tabulka 8: Úspěšnost sítí navržených v rámci prvního experimentu.

Měření	**Přesnost sítě [%]**		
	Net-01	Net-02	Net-03
#1	76,67	89,72	83,76
#2	87,77	78,03	86,89
#3	85,49	83,53	83,15
#4	83,59	75,59	83,51
#5	84,55	82,43	82,83
Průměr	83,61	82,22	**84,03**

Nejlépe se podařilo natrénovat síť Net-03, která dosáhla přesnosti v průměru 84,03%. Tato síť má velmi podobnou architekturu jako Net-02. Jediným rozdílem je prohození batch normalizace a aktivační funkce ReLU. Toto zjištění bylo překvapující, jelikož v práci [35] se shodují, že batch normalizace by měla následovat až za aktivační funkcí, aby přepočítala data na nulovou střední hodnotu s rozptylem 1. Volání aktivační funkce po batch normalizaci by mělo za následek zahození poloviny dat. Průběh trénování obou sítí byl proto zanesen do grafu na obrázku 26 pro jejich porovnání.



Obrázek 26: Porovnání přesnosti a chyby v průběhu trénování pro sítě Net-02 a Net-03.

Přesnost na trénovací množině úspěšnější sítě Net-03 roste v průběhu trénování pomaleji. Zároveň můžeme pozorovat, že jakmile přesnost na trénovací množině dosáhne hodnoty okolo 99%, přesnost na testovací množině již neroste. Vložením aktivační funkce ReLU za batch normalizaci, tedy zahozením poloviny rozsahu hodnot má pravděpodobně za následek pomalejší učení sítě a tedy vyšší výslednou přesnost.

Tuto teorii podkládá i fakt, že síť Net-01 s vrstvou dropout (parametrem 0,5) má vyšší úspěšnost než síť Net-02, jelikož zahozením poloviny hodnot docílíme také pomalejšího učení. Net-01 je zároveň jedinou sítí s dropout vrstvou, kterou se podařilo natrénovat na úspěšnost vyšší než 80%. Přidáním dalších dropout vrstev došlo většinou k zahození příliš velkého počtu informací a síť se po 10 000 krocích přestala zlepšovat a po 30 000 krocích bylo trénování ručně přerušeno.

Dalším zjištěním byla vyšší úspěšnost AVG pooling vrstev oproti MAX pooling vrstev. Důvod je celkem prostý. Funkce MAX s velikostí filtru 2×2 a posunu o 2 přenese na výstup pouze jednu ze čtyř hodnot. Funkce AVG spočítá průměr hodnot a přestože jsou hodnoty zredukovány pouze do jednoho čísla, obsahuje toto číslo více informací. Přihlédneme-li na malý rozměr vstupní matice příznaků, je tato vlastnost více než žádaná.

6.4.2 Experiment 2

Druhý experiment měl za cíl využít znalosti získané v rámci prvního experimentu a navrhnout hlubší síť s pomocí reziduálních bloků sítě ResNet popsané v podsekci 4.3.5. V rámci experimentu bylo vytvořeno přes 30 modifikací sítě, které se většinou lišily počtem reziduálních bloků a počtem konvolucí v rámci konvoluční vrstvy. Motivací za návrhem modifikované sítě ResNet byla její určitá tolerance vůči nevhodně zvolené hloubce sítě.

Tabulka 9: Výběr navržených architektur inspirovaných sítí ResNet.

#	**Net-04**	**Net-05**	**Net-06**	**Net-07**
1	Konvoluce 64 @ 3 × 3 / 1	Konvoluce 64 @ 3 × 3 / 1	Konvoluce 64 @ 3 × 3 / 1	Konvoluce 16 @ 3 × 3 / 1
2	BN + ReLU	BN + ReLU	BN + ReLU	BN + ReLU
3	Reziduální blok 64 @ 3 × 3 / 1	Reziduální blok 64 @ 3 × 3 / 1	3× Reziduální blok 16 @ 3 × 3 / 1	3× Reziduální blok 16 @ 3 × 3 / 1
4	AVG Pool 2 × 2 / 2	AVG Pool 2 × 2 / 2	Reziduální blok 32 @ 3 × 3 / 1	Reziduální blok 32 @ 3 × 3 / 1
5	Reziduální blok 128 @3 × 3 / 1	Reziduální blok 128 @3 × 3 / 1	AVG Pool 2 × 2 / 2	AVG Pool 2 × 2 / 2
6	Globální AVG Pool	AVG Pool 2 × 2 / 2	3× Reziduální blok 32 @ 3 × 3 / 1	3× Reziduální blok 32 @ 3 × 3 / 1
7	Plně spojená 128	Reziduální blok 128 @ 3 × 3 / 1	Reziduální blok 64 @ 3 × 3 / 1	Reziduální blok 64 @ 3 × 3 / 1
8	-	Globální AVG Pool	AVG Pool 2 × 2 / 2	AVG Pool 2 × 2 / 2
9	-	Plně spojená 128	Reziduální blok 128 @ 3 × 3 / 1	Reziduální blok 128 @ 3 × 3 / 1

Tabulka 9: Výběr navržených architektur inspirovaných sítí ResNet.

#	Net-04	Net-05	Net-06	Net-07
10	-	-	4× Reziduální blok 128 @ 3 × 3 / 1	4× Reziduální blok 128 @ 3 × 3 / 1
11	-	-	Globální AVG Pool	Globální AVG Pool
12	-	-	Plně spojená 128	Plně spojená 128

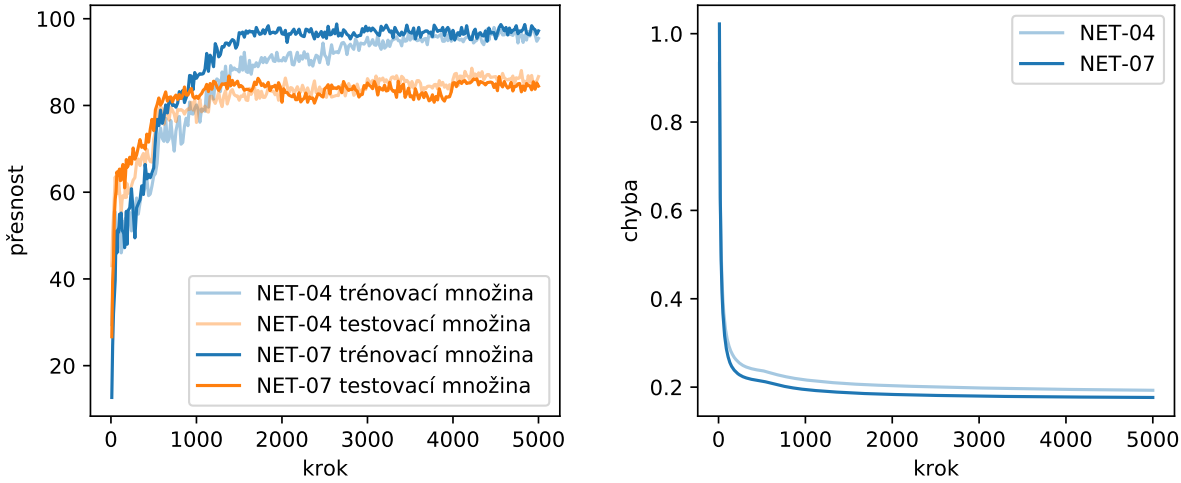
Je na místě připomenout, že každý reziduální blok obsahuje 2 konvoluční vrstvy a po každé konvoluční vrstvě se provádí batch normalizace a následně ReLU. U popisu architektur Net-06 a Net-07 v tabulce 9 byly reziduální bloky se stejnými parametry sjednoceny do stejné kolonky z důvodu kratšího zápisu. Počet reziduálních bloků za sebou určuje násobitel vložený před název vrstvy.

Při návrhu hlubších sítí bylo potřeba dát si pozor na počet pooling vrstev. Při dvou operacích pool a vstupním obrazu s rozlišením 36×20 se redukuje aktivační maska na rozměr 9×5. Proto i síť Net-07 tvořena 85 vrstvami (z toho 54 konvolučních), obsahuje pouze 2 pooling vrstvy. Trénování probíhalo stejně jako v prvním experimentu s výjimkou nastavení mini-batch parametrů. Z důvodu příliš hluboké sítě se stejně velký počet dat nevlezl do paměti počítače, proto byly parametry upraveny na hodnoty $(P, S) = (30, 5)$.

Tabulka 10: Úspěšnost sítí navržených v rámci druhého experimentu.

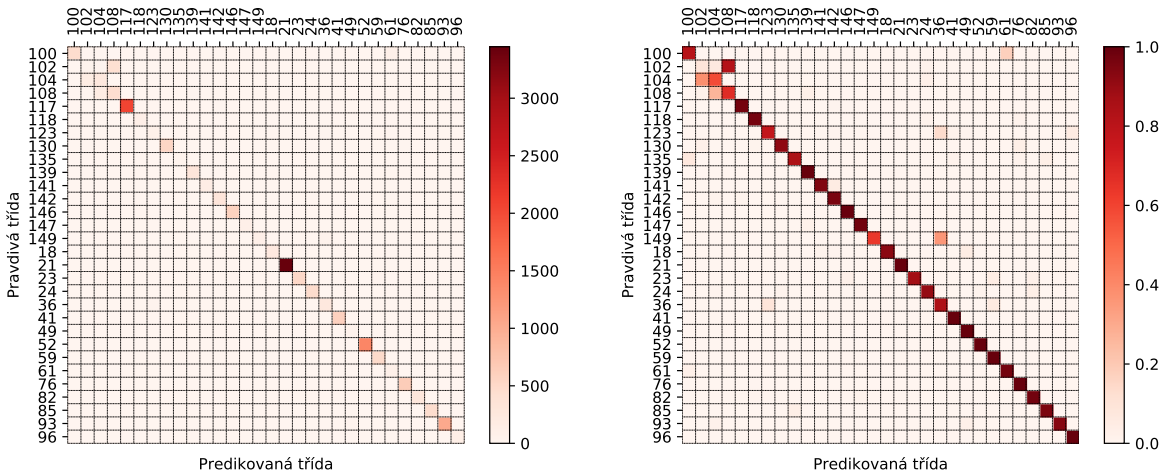
Měření	**Přesnost sítě [%]**			
	Net-04	Net-05	Net-06	Net-07
#1	87,25	88,48	88,57	85,00
#2	68,62	91,07	78,86	87,16
#3	89,82	79,19	93,07	85,21
#4	76,33	86,54	76,89	91,19
#5	88,53	82,07	85,42	87,91
Průměr	82,11	85,47	84,56	**87,30**

Z počátku byly vytvářeny méně hluboké sítě typu Net-04, která má pouze dva reziduální bloky. Důvodem byla teorie, že příliš mnoho konvolucí by způsobilo, že i síť s reziduálními bloky by se na tak malém obrazu nedokázala dostatečně natrénovat. Postupné přidávání reziduálních bloků však dosahovalo lepších a lepších výsledků, které jsou srovnatelné se sítěmi z prvního experimentu. Z tabulky 10 je patrné, že stropem v přidávání reziduálních bloků byla síť Net-06, která dosáhla horších výsledků než její předchůdci. Po několika změnách v počtu konvolucí vznikla síť Net-07. Tuto síť se podařilo natrénovat nejlépe a to s přesností v průměru 87,30%.



Obrázek 27: Porovnání přesnosti a chyby v průběhu trénování pro sítě Net-04 a Net-07.

Porovnáme-li průběh trénování se sítí Net-04, můžeme vidět, že přesnost na trénovací množině dosáhne rychleji vyšší hodnoty. Oproti podobnému případu sítí Net-02 a Net-03 z prvního experimentu, kdy tato vlastnost byla na škodu, zde pozorujeme také výrazně nižší chybu sítě Net-04 v průběhu celého trénování.



Obrázek 28: Vizualizace konfúzní matice s absolutními hodnotami (vlevo) a procentuální (vpravo) získané z testování sítě Net-07 v rámci čtvrtého testu.

Konfúzní matice čtvrtého testu sítě Net-07 na obrázku 28 ukazuje, že síť zaměňuje osobu 104 jako 102, osobu 102 jako 108 a osobu 149 jako 36. Výběr snímků těchto osob je na obrázku 29. Osoby 102, 104 a 108 jsou oblečeny do bílého trička, dokonce obě mají tašku přes rameno. Pokud se podíváme na druhou situaci, černý deštník osoby 149 na spodním snímku překrývá

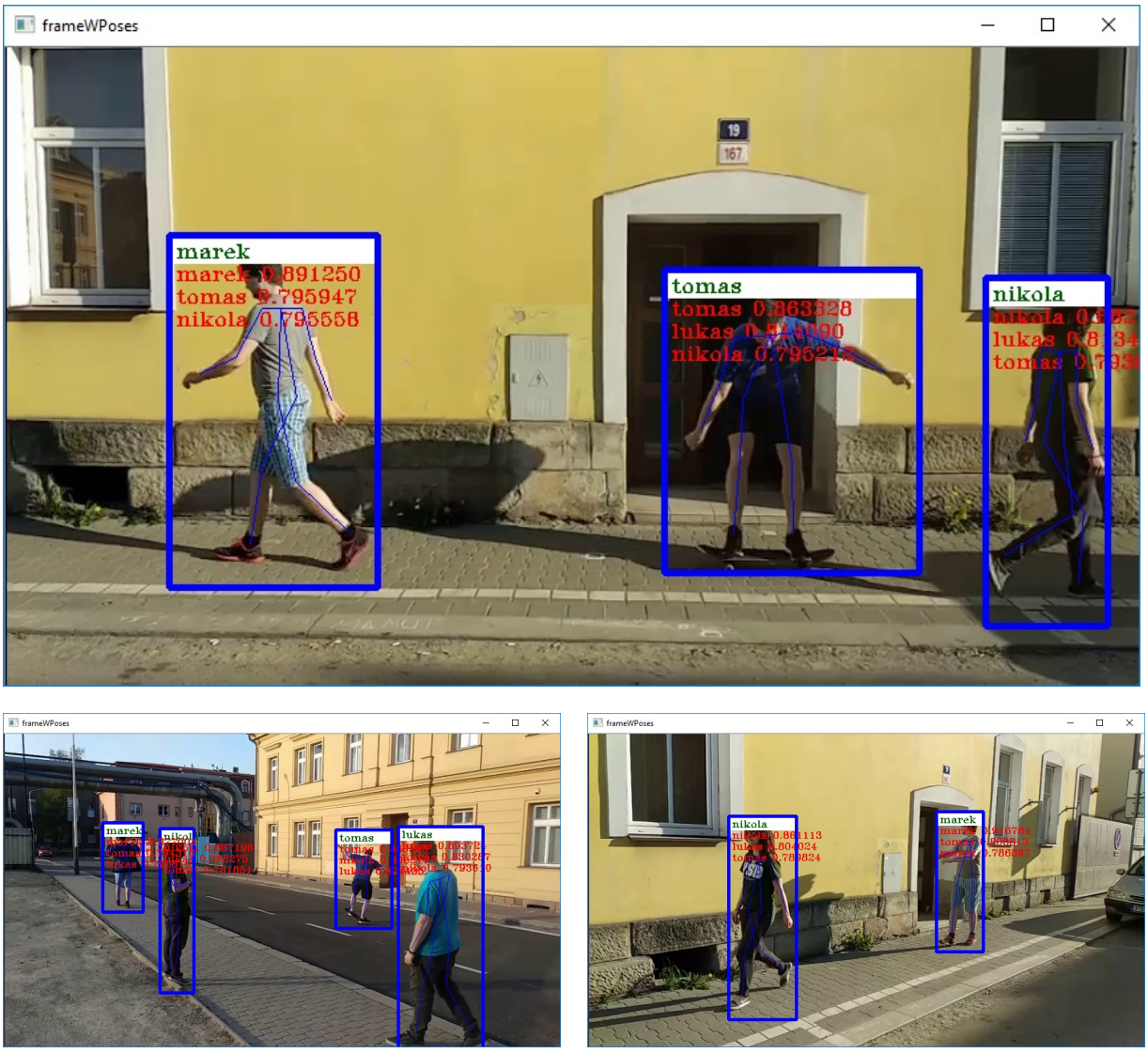
nohy a může vypadat jako černé kalhoty osoby 36. Zvolená metoda pro opětovnou identifikaci v těchto případech selhává.

Obrázek 29: Snímky nesprávně rozeznaných osob sítí Net-07.

Druhý experiment prokázal úspěšnost reziduálních bloků v rámci konvolučních sítí. Síť Net-07 je nejlépe natrénovanou sítí v rámci celého projektu a oproti nejlepší sítí Net-04 z prvního experimentu dosahuje o 3,27% vyšší úspěšnosti. Díky své vysoké úspěšnosti je tato síť použita v navržené aplikaci pro získání deskriptoru detekované osoby.

7 Testování modelového případu

Pro ověření funkčnosti navrženého systému byla vytvořena jednoduchá konzolová aplikace. Tato aplikace byla testována na modelovém případu, kdy vstupem je sekvence snímků a výstupem seznam osob nacházejících se každém ze snímku. Vstupem může být přímý vstup z kamery, video nebo sekvence obrázku a výstupní soubor je ve formátu JSON. Modelový případ má napodobit situaci, kdy chceme nalézt podezřelou osobu na více kamerových záznamech.



Obrázek 30: Ukázka aplikace navrženého systému na testovací sekvenci B. Algoritmus dokáže identifikovat osoby v různé póze a napříč různými záběry kamery.

Testování probíhalo na dvou sekvencích. První sekvence *A* obsahuje celkem 30 různých osob z datasetu MARS a je popsána v sekci 5.1. Zároveň pro tento účel byla natočena nová sekvence *B* zabírající různé scény se stejnými osobami. Jedno z těchto videí je pořízeno s časovým odstu-

pem, aby byla scéna zachycena i při jiném osvětlení. Tabulka 11 popisuje jednotlivé sekvence. V případě, že na snímku nebyla detekována osoba nebo bylo detekováno osob více, nebylo možné ověřit predikci a snímek byl ignorován (nelze detekovat). V případě, že byla póza úspěšně detekována nebo bylo možné zrekonstruovat matici příznaků, je póza označena jako validní a je provedena predikce identity osoby. V opačném případě je póza neplatná.

Tabulka 11: Vlastnosti testovacích sekvencí A a B. Přehled kolik je v sekvenci zaznamenáno celkem detekcí, na kolika z nich je validní nebo neplatná póza a na kolika nebylo možné provézt predikci z důvodu více či méně než jedné osoby na snímku.

Sekvence	**Počet osob**	**Celkem**	**Validní póza**	**Neplatná póza**	**Nelze detekovat**
A	30	19 648	13 947	2 238	3 463
B	4	2 436	2 152	140	144

Dalo by se očekávat, že čím více bude mít aplikace referenčních snímků jedné osoby, tím bude její úspěšnost vyšší. Při testování proto byly vybírány postupně 1, 2 a 3 referenční snímky. Výběr referenčních snímků probíhal v každém měření náhodně, aby bylo možné vidět, jaký vliv má výběr referenčního snímku na úspěšnost. Výsledky pro obě sekvence jsou prezentovány v tabulce 12.

Tabulka 12: Úspěšnost predikce v závislosti na počtu referenčních snímků pro sekvence A a B.

(a) Sekvence A

Počet ref. snímků	**1**	**2**	**3**
Měření	**Úspěšnost [%]**		
#1	79,34	90,69	93,44
#2	82,54	92,09	88,67
#3	85,31	91,31	93,97
#4	90,60	91,23	90,17
#5	82,32	88,29	95,28
Průměr	84,02	90,72	92,31

(b) Sekvence B

Počet ref. snímků	**1**	**2**	**3**
Měření	**Úspěšnost [%]**		
#1	96,36	95,86	98,76
#2	67,63	83,61	95,86
#3	96,52	99,59	96,94
#4	72,35	95,53	94,87
#5	97,27	97,19	97,52
Průměr	86,03	94,35	96,79

Testování ověřilo očekávaný výsledek, že větší počet referenčních snímků má pozitivní vliv na úspěšnou predikci aplikace. Vyšší úspěšnost sekvence B je dána menším počtem různých osob (4 osoby oproti 30) a dá se předpokládat, že s přibývajícím počtem lidí se bude úspěšnost aplikace dále snižovat. Výsledky také ukazují důležitost výběru referenčního snímku. Ve druhém měření s jedním referenčním snímkem na sekvenci B byly vybrány špatné referenční snímky a úspěšnost dosáhla pouhých 67,63%. Na obrázku 31 je výběr referenčních snímků porovnán s výběrem pro páté měření, kdy úspěšnost 97,27% byla zároveň nejvyšší.

(a) úspěšnost 67,63%

(b) úspěšnost 97,27%

Obrázek 31: Porovnání referenčních snímků sekvence B s nízkou a vysokou úspěšností. Referenční snímek první osoby zprava je důvodem rozdílu úspěšností.

Obě sady referenčních snímků jsou na první pohled velmi podobné. Nejvíce se liší referenční snímek první osoby zprava. Ten je také hlavní příčinou rozdílu v úspěšnosti. Výběr příliš tmavého referenčního snímku způsobil nízkou úspěšnost, protože nedostatečné osvětlení znehodnotilo příznaky potřebné k predikci. Bylo také potřeba provést rekonstrukci matice příznaků, protože na obrázku není vidět levá ruka a pravá noha. Kvůli sjednocení velikostí jednotlivých snímků už nejde vidět, že snímek má rozlišení pouze 40×150 pixelů, což také může vést ke ztrátě příliš mnoha detailů potřebných k opětovné identifikaci. Řešením problému může být výběr kamery s kvalitnějším záznamem a vyšším rozlišením, ale tím problém jenom oddálíme. Dalším řešením pak může být zajištění dostatečného osvětlení scény nebo výběr více referenčních snímků.

Tabulka 13: Čas potřebný k opětovné identifikaci osob z jednoho snímku s rozlišením 854×480.

	Čas [ms]			
Počet osob	**Detekce**	**Sestavení matice příznaků**	**Klasifikace**	**Celkem**
1	123	5	1	129
3	124	13	1	138

Tabulka 13 udává čas potřebný k opětovné identifikaci osob v obraze. Čas k výpočtu deskriptoru konvoluční sítí a následné klasifikaci osoby je zanedbatelný. Extrakce příznaků a sestavení matice příznaků zabere určitý čas a s narůstajícím počtem osob se zvyšuje, protože je potřeba počítat novou matici pro každou osobu. Nejvíce času zabere detekce pózy osoby knihovnou OpenPose. Jedná se téměř o 90% z celkového času v případě tří osob v obraze. Tento čas se s narůstajícím počtem osob v obraze nezvyšuje.

8 Závěr

Náplní práce je sledování pohybu osob ve vymezeném prostoru. Jedná se o problém opětovné identifikace osoby, tedy úlohy rozpoznat osobu v obraze, která již byla dříve pozorována. Obraz je často v nízkém rozlišení, proto se při řešení spoléhá pouze na rozpoznání vzhledu oblečení. Navržené metody většinou pracují se stavbou lidského těla a rozdělují obraz na důležité části. Kromě nízkého rozlišení je potřeba brát také v úvahu různé světelné podmínky a pózu osoby.

V teoretické části byly popsány dosavadní přístupy detekce a identifikace osoby a na základě jejich poznatků byla navržena struktura systému pro opětovnou identifikaci osoby. Systém byl navržen tak, aby z obrazu získal popis osoby (deskriptor) a na základě podobnosti deskriptoru určil její identitu. Namísto běžných přístupů pro detekci osoby v obraze byla použita knihovna OpenPose, která dokáže určit pózu osoby. Tato póza je použita k sestavení matice příznaků. Pro získání deskriptoru z matice příznaků byla použita konvoluční neuronová síť. Pro zasvěcení do problematiky konvolučních sítí bylo nutné nastudovat princip jejich fungování, účel jednotlivých vrstev a v čem jsou jednotlivé architektury přelomové.

Praktická část se zabývá implementací navrženého postupu. Vhodnou testovací množinou je dataset MARS, který obsahuje dostatečný počet unikátních osob zachycených v různých pózách a na vícero kamerách. Ne všechny snímky však byly pro testování systému vyhovující, proto bylo nutné dataset filtrovat. Dalším důvodem užšího výběru datasetu bylo časově náročné zpracování všech dat. V práci je popsáno jak probíhá sestavení matice příznaků z detekované pózy a byla představena metoda pro rekonstrukci příznaků v případě, že póza není správně detekována. Samostatná kapitola byla věnována konvolučním sítím. Nejedná se o standardní problém klasifikace, tudíž hledání správné architektury a způsobu jak síť trénovat zabralo nejvíce času a spoustu návrhů kvůli špatným výsledkům nebylo možné použít. Práce detailně popisuje postup při trénování. Nejprve bylo pro rychlejší trénování potřeba celou testovací množinu zpracovat na matice příznaků. V tabulkách a grafech bylo shrnuto, jak nastavení jednotlivých parametrů ovlivnilo průběh trénování. Ukázalo se, že v každém kroku trénování je lepší vybírat jiný referenční snímek pro test, protože takový test má větší vypovídající hodnotu. Výběr nejlepší architektury byl shrnut do dvou experimentů, kdy nejlepšího výsledku dosáhla síť postavená na reziduálních blocích.

Byl představen modelový případ, který ověřil funkčnost implementované aplikace na dvou testovacích sekvencích. Aplikace pro každý snímek vstupní sekvence predikovala identitu detekovaných osob na základě jejich referenčních snímků. Seznam predikcí pro každý snímek pak uložila na disk. Testování pro 30 různých identit dopadlo poměrně úspěšně, ale poukázalo na klesající úspěšnost s narůstajícím počtem predikovaných osob. To může být způsobeno zvyšující se šancí pro výskyt podobně oblečených lidí a natrénovaná síť je nedokáže dostatečně rozlišit. Tento problém by mohla řešit větší trénovací sada, tu však z časových důvodů nebylo možné vytvořit. Snímky, které jsou pořízeny v příliš odlišném osvětlení scény, než je na vstupní sekvenci, nedosahují takové úspěšnosti, zvláště pak v šeru či tmě. Řešením se nabízí více refe-

renčních snímků z různého osvětlení. Testy prokázaly, že s větším počtem referenčních snímků dosahuje aplikace věští úspěšnosti. V případě jednoho referenčního snímku je pak důležité klást dostatečnou pozornost na jeho výběr.

Implementovány algoritmus vyžaduje, aby byl v testovací sekvenci k dispozici referenční snímek pro každou osobu. To znamená, že se v obraze nesmí vyskytovat neznámá osoba. Deskriptory různých snímků stejné osoby totiž nemají konstantní vzdálenost a jsou ovlivněny různými faktory. Nelze tak určit, kdy je vzdálenost vysoká z důvodu změny pózy osoby nebo jiného osvětlení scény, a kdy je detekována neznámá osoba. Navazující práce by se mohla zaměřit na nalezení metody pro rozpoznání, že pro detekovanou osobu není k dispozici referenční snímek.

Seznam literatury

[1] ZHENG, Liang; YANG, Yi; HAUPTMANN, Alexander G. Person Re-identification: Past, Present and Future. *CoRR*. 2016, roč. abs/1610.02984. Dostupné z arXiv: 1610.02984.

[2] YANG, M.; YU, K. Real-time clothing recognition in surveillance videos. In: *2011 18th IEEE International Conference on Image Processing*. 2011, s. 2937–2940. ISSN 1522-4880. Dostupné z DOI: 10.1109/ICIP.2011.6116276.

[3] THAPAR, Daksh; AGGARWAL, Divyansh; AGARWAL, Punjal; NIGAM, Aditya. VGR-Net: A View Invariant Gait Recognition Network. *CoRR*. 2017, roč. abs/1710.04803. Dostupné z arXiv: 1710.04803.

[4] STAUFFER, Chris; E. L. GRIMSON, W. Adaptive background mixture models for real-time tracking. 2007, roč. 2.

[5] DALAL, N.; TRIGGS, B. Histograms of oriented gradients for human detection. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. 2005, sv. 1, 886–893 vol. 1. ISSN 1063-6919. Dostupné z DOI: 10.1109/CVPR.2005.177.

[6] GIRSHICK, Ross B.; DONAHUE, Jeff; DARRELL, Trevor; MALIK, Jitendra. Rich feature hierarchies for accurate object detection and semantic segmentation. *CoRR*. 2013, roč. abs /1311.2524. Dostupné z arXiv: 1311.2524.

[7] GIRSHICK, Ross B. Fast R-CNN. *CoRR*. 2015, roč. abs/1504.08083. Dostupné z arXiv: 1504.08083.

[8] REN, Shaoqing; HE, Kaiming; GIRSHICK, Ross B.; SUN, Jian. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *CoRR*. 2015, roč. abs/1506.01497. Dostupné z arXiv: 1506.01497.

[9] ZHENG, Liang; BIE, Zhi; SUN, Yifan; WANG, Jingdong; SU, Chi; WANG, Shengjin; TIAN, Qi. *European Conference on Computer Vision*. MARS: A Video Benchmark for Large-Scale Person Re-identification. 2007.

[10] YANG, Yang; YANG, Jimei; YAN, Junjie; LIAO, Shengcai; YI, Dong; LI, Stan Z. Salient Color Names for Person Re-identification. In: *ECCV*. 2014.

[11] GRAY, Doug; BRENNAN, Shane; TAO, Hai. Evaluating appearance models for recognition, reacquisition, and tracking. In: *In IEEE International Workshop on Performance Evaluation for Tracking and Surveillance, Rio de Janeiro*. 2007.

[12] YI, Dong; LEI, Zhen; LI, Stan Z. Deep Metric Learning for Practical Person Re-Identification. *CoRR*. 2014, roč. abs/1407.4979. Dostupné z arXiv: 1407.4979.

[13] AHMED, Ejaz; JONES, Michael J.; MARKS, Tim K. An improved deep learning architecture for person re-identification. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, s. 3908–3916.

[14] ZHANG, Wei; HU, Shengnan; LIU, Kan. Learning Compact Appearance Representation for Video-based Person Re-Identification. *CoRR*. 2017, roč. abs/1702.06294. Dostupné z arXiv: 1702.06294.

[15] MA, Xiaolong; ZHU, Xiatian; GONG, Shaogang; XIE, Xudong; HU, Jianming; LAM, Kin-Man; ZHONG, Yisheng. Person re-identification by unsupervised video matching. *Pattern Recognition*. 2017, roč. 65, s. 197–210. ISSN 0031-3203. Dostupné z DOI: https://doi.org/10.1016/j.patcog.2016.11.018.

[16] LIU, Ziwei; LUO, Ping; QIU, Shi; WANG, Xiaogang; TANG, Xiaoou. DeepFashion: Powering Robust Clothes Recognition and Retrieval with Rich Annotations. In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[17] YANG, M.; YU, K. Real-time clothing recognition in surveillance videos. In: *2011 18th IEEE International Conference on Image Processing*. 2011, s. 2937–2940. ISSN 1522-4880. Dostupné z DOI: 10.1109/ICIP.2011.6116276.

[18] KALANTIDIS, Y.; KENNEDY, L.; LI, L.-J. Getting the Look: Clothing Recognition and Segmentation for Automatic Product Suggestions in Everyday Photos. In: *in Proceedings of International Conference on Multimedia Retrieval (ICMR) (ICMR 2013)*. Dallas, TX: ACM, 2013.

[19] OJALA, T.; PIETIKAINEN, M.; MAENPAA, T. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2002, roč. 24, č. 7, s. 971–987. ISSN 0162-8828. Dostupné z DOI: 10.1109/TPAMI.2002.1017623.

[20] BRADSKI, G. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*. 2000.

[21] KING, Davis E. Dlib-ml: A Machine Learning Toolkit. *Journal of Machine Learning Research*. 2009, roč. 10, s. 1755–1758.

[22] CAO, Zhe; SIMON, Tomas; WEI, Shih-En; SHEIKH, Yaser. Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. In: *CVPR*. 2017.

[23] SIMON, Tomas; JOO, Hanbyul; MATTHEWS, Iain; SHEIKH, Yaser. Hand Keypoint Detection in Single Images using Multiview Bootstrapping. In: *CVPR*. 2017.

[24] WEI, Shih-En; RAMAKRISHNA, Varun; KANADE, Takeo; SHEIKH, Yaser. Convolutional pose machines. In: *CVPR*. 2016.

[25] NVIDIA CORPORATION. *NVIDIA CUDA C Programming Guide*. 2010. Version 3.2.

[26] RIZA ALP GÜLER Natalia Neverova, Iasonas Kokkinos. DensePose: Dense Human Pose Estimation In The Wild. *arXiv*. 2018.

[27] MEHTA, Dushyant; SRIDHAR, Srinath; SOTNYCHENKO, Oleksandr; RHODIN, Helge; SHAFIEI, Mohammad; SEIDEL, Hans-Peter; XU, Weipeng; CASAS, Dan; THEOBALT, Christian. VNect: Real-time 3D Human Pose Estimation with a Single RGB Camera. In: 2017, sv. 36. Č. 4. Dostupné z DOI: 10.1145/3072959.3073596.

[28] WALKER, Jacob; MARINO, Kenneth; MULAM, Harikrishna; HEBERT, Martial. The Pose Knows: Video Forecasting by Generating Pose Futures. 2017.

[29] JIA, Yangqing; SHELHAMER, Evan; DONAHUE, Jeff; KARAYEV, Sergey; LONG, Jonathan; GIRSHICK, Ross; GUADARRAMA, Sergio; DARRELL, Trevor. Caffe: Convolutional Architecture for Fast Feature Embedding. *arXiv preprint arXiv:1408.5093*. 2014.

[30] YOSINSKI, Jason; CLUNE, Jeff; NGUYEN, Anh Mai; FUCHS, Thomas J.; LIPSON, Hod. Understanding Neural Networks Through Deep Visualization. *CoRR*. 2015, roč. abs/1506.06579. Dostupné z arXiv: 1506.06579.

[31] IANDOLA, Forrest N. Exploring the Design Space of Deep Convolutional Neural Networks at Large Scale. *CoRR*. 2016, roč. abs/1612.06519. Dostupné z arXiv: 1612.06519.

[32] SPRINGENBERG, Jost Tobias; DOSOVITSKIY, Alexey; BROX, Thomas; RIEDMILLER, Martin A. Striving for Simplicity: The All Convolutional Net. *CoRR*. 2014, roč. abs/1412.6806. Dostupné z arXiv: 1412.6806.

[33] KRIZHEVSKY, Alex. Learning Multiple Layers of Features from Tiny Images. 2012.

[34] RUSSAKOVSKY, Olga et al. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*. 2015, roč. 115, č. 3, s. 211–252. Dostupné z DOI: 10.1007/s11263-015-0816-y.

[35] IOFFE, Sergey; SZEGEDY, Christian. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *CoRR*. 2015, roč. abs/1502.03167. Dostupné z arXiv: 1502.03167.

[36] LECUN, Y.; BOTTOU, L.; BENGIO, Y.; HAFFNER, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*. 1998, roč. 86, č. 11, s. 2278–2324. ISSN 0018-9219. Dostupné z DOI: 10.1109/5.726791.

[37] KRIZHEVSKY, Alex; SUTSKEVER, Ilya; HINTON, Geoffrey E. ImageNet Classification with Deep Convolutional Neural Networks. In: PEREIRA, F.; BURGES, C. J. C.; BOTTOU, L.; WEINBERGER, K. Q. (ed.). *Advances in Neural Information Processing Systems 25*. Curran Associates, Inc., 2012, s. 1097–1105. Dostupné také z: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

[38] SIMONYAN, Karen; ZISSERMAN, Andrew. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR*. 2014, roč. abs/1409.1556. Dostupné z arXiv: 1409.1556.

[39] SZEGEDY, Christian; LIU, Wei; JIA, Yangqing; SERMANET, Pierre; REED, Scott E.; ANGUELOV, Dragomir; ERHAN, Dumitru; VANHOUCKE, Vincent; RABINOVICH, Andrew. Going Deeper with Convolutions. *CoRR*. 2014, roč. abs/1409.4842. Dostupné z arXiv: 1409.4842.

[40] LIN, Min; CHEN, Qiang; YAN, Shuicheng. Network In Network. *CoRR*. 2013, roč. abs/1312.4400. Dostupné z arXiv: 1312.4400.

[41] HE, Kaiming; ZHANG, Xiangyu; REN, Shaoqing; SUN, Jian. Deep Residual Learning for Image Recognition. *CoRR*. 2015, roč. abs/1512.03385. Dostupné z arXiv: 1512.03385.

[42] HU, Jie; SHEN, Li; SUN, Gang. Squeeze-and-Excitation Networks. *CoRR*. 2017, roč. abs/1709.01507. Dostupné z arXiv: 1709.01507.

[43] DEHGHAN, A.; ASSARI, S. M.; SHAH, M. GMMCP tracker: Globally optimal Generalized Maximum Multi Clique problem for multiple object tracking. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, s. 4091–4099. ISSN 1063-6919. Dostupné z DOI: 10.1109/CVPR.2015.7299036.